Compresión sin pérdida de imágenes satelitales multiespectrales e hiperespectrales

Acevedo, Daniel

2011

Cita tipo APA:
Acevedo, Daniel. (2011). Compresión sin pérdida de imágenes satelitales multiespectrales e hiperespectrales. Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires.

Cita tipo Chicago:
Acevedo, Daniel. "Compresión sin pérdida de imágenes satelitales multiespectrales e hiperespectrales". Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires. 2011.

UNIVERSIDAD DE BUENOS AIRES
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Compresión sin Pérdida de Imágenes Satelitales Multiespectrales e Hiperespectrales

Tesis presentada para optar al título de Doctor de la Universidad de Buenos Aires en el área Ciencias de la Computación

Daniel Acevedo

Directora: Dra. Ana M. C. Ruedin
Consejero de estudios: Dra. Marta Mejail

Buenos Aires, 2011.

Lossless Compression of Satellite Multispectral and Hyperspectral Images

Abstract

In this thesis, new lossless compression techniques aiming at reducing the size of storage of satellite images are presented. Two type of images are considered: multispectral and hyperspectral.

For multispectral images, a nonlinear lossless compressor that exploits both intraband and interband correlations is developed. The compressor is based on a wavelet transform that maps integers into integers, applied to tiles of the image. Different models for statistical dependencies of wavelet detail coefficients are proposed and analyzed. Wavelet coefficients belonging to the fine detail subbands are successfully modelled as an affine combination of neighboring coefficients and the coefficient at the same location in the previous band, as long as all these coefficients belong to the same landscape. This model is used to predict wavelet coefficients by means of already coded coefficients. Lloyd-Max quantization is used to extract class information, which is used in the prediction and later used as a conditioning context to encode prediction errors with an adaptive arithmetic coder. The band order affects the accuracy of predictions: a new mechanism is proposed for ordering the bands, based on the wavelet detail coefficients of the 2 finest levels. The results obtained outperform 2D lossless compressors such as PNG, JPEG-LS, SPIHT and JPEG2000 and other 3D lossless compressors such as SLSQ-OPT, differential JPEG-LS, JPEG2000 for color images and 3D-SPIHT. Our method has random access capability, and can be applied for lossless compression of other kinds of volumetric data.

For hyperspectral images, state-of-the-art algorithms LUT and LAIS-LUT proposed for lossless compression, exploit high spectral correlations in these images, and use lookup tables to perform predictions. However, there are cases where their predictions are not accurate. In this thesis a modification based also on look-up tables is proposed, giving these tables different degrees of confidence, based on the local variations of the scaling factor. Our results are highly satisfactory and outperform both LUT and LAIS-LUT methods.

Two lossless compressors have been designed for two different kinds of satellite images having different properties, namely, different spectral resolution, spatial resolution, and bitdepth, as well as different spectral and spatial correlations. In each case, the compressor exploits these properties to increase compression ratios.

Keywords: Satellite images, multispectral, hyperspectral, lossless compression, wavelet transform, prediction.

COMPRESIÓN SIN PÉRDIDA DE IMÁGENES SATELITALES MULTIESPECTRALES E HIPERESPECTRALES

Resumen

En esta tesis se presentan nuevas técnicas de compresión sin pérdida tendientes a reducir el espacio de almacenamiento requerido por imágenes satelitales. Dos tipos principales de imágenes son tratadas: multiespectrales e hiperespectrales.

En el caso de imágenes multiespectrales, se desarrolló un compresor no lineal que explota tanto las correlaciones intra como interbanda presentes en la imagen. Éste se basa en la transformada wavelet de enteros a enteros y se aplica sobre bloques no solapados de la imagen. Diferentes modelos para las dependencias estadísticas de los coeficientes de detalle de la transformada wavelet son propuestos y analizados. Aquellos coeficientes que se encuentran en las subbandas de detalle fino de la transformada son modelados como una combinación afín de coeficientes vecinos y coeficientes en bandas adyacentes, sujetos a que se encuentren en la misma clase. Este modelo se utiliza para generar predicciones de otros coficientes que ya fueron codificados. La información de clase se genera mediante la cuantización LloydMax, la cual también se utiliza para predecir y como contextos de condicionamiento para codificar los errores de predicción con un codificador aritmético adaptativo. Dado que el ordenamiento de las bandas también afecta la precisión de las predicciones, un nuevo mecanismo de ordenamiento es propuesto basado en los coeficientes de detalle de los últimos dos niveles de la transformada wavelet. Los resultados obtenidos superan a los de otros compresores 2D sin pérdida como PNG, JPEG-LS, SPIHT y JPEG2000, como también a otros compresores 3D como SLSQ-OPT, JPEG-LS diferencial y JPEG2000 para imágenes a color y 3D-SPIHT. El método propuesto provee acceso aleatorio a partes de la imagen, y puede aplicarse para la compresión sin pérdida de otros datos volumétricos.

Para las imágenes hiperespectrales, algoritmos como LUT o LAIS-LUT que revisten el estado del arte para la compresión sin pérdida para este tipo de imágenes, explotan la alta correlación espectral de estas imágenes y utilizan tablas de lookup para generar predicciones. A pesar de ello, existen casos donde las predicciones no son buenas. En esta tesis, se propone una modificación a estos algoritmos de lookup permitiendo diferentes niveles de confianza a las tablas de lookup en base a las variaciones locales del factor de escala. Los resultados obtenidos son altamente satisfactorios y mejores a los de LUT y LAIS-LUT.

Se han diseñado dos compresores sin pérdida para dos tipos de imágenes satelitales, las cuales tienen distintas propiedades, a saber, diferente resolución espectral, espacial y radiométrica, y también de diferentes correlaciones espectrales y espaciales. En cada caso, el compresor explota estas propiedades para incrementar las tasas de compresión.

Palabras clave: imágenes satelitales, multiespectral, hiperespectral, compresión sin pérdida, transformada wavelet, predicción.

Agradecimientos

A mi directora, Dra. Ana Ruedin, a quien le debo la mayor parte de mi formación como investigador y la obtención de este título, y con quien he cultivado una gran amistad todos estos años.

A mi familia, a quienes amo y son mi gran soporte.

A mis amigos de la vida y del DC: a María Elena Buemi por su gran amistad y por transitar juntos todos estos años. A Marta Mejail por su infinita generosidad y al resto de las personas que forman el grupo de Procesamiento de Imágenes del DC (Julio Jacobo, Norberto Goussies, Alexandra Diehl, Mariano Tepper, Pachi, etc). A Rosana Matuk y Rocío Romero, grandes amigas del DC. A Javier Quinteros, Leticia Seijas, Esther De Ves, Isabel Méndez, Paula Zabala, Irene Loiseau y Aida Interlandi, gente muy valiosa que conocí estos años en el DC. Al resto de la gente del DC con quienes transité este tramo de mi formación; seguramente me olvido de muchos, pero ustedes saben quiénes son.

A Nelson Zeljkovich, Franco Balcarce, Sebastián Lloret, Augusto Almada, Diego Dundic, Mariana Gallo, Agustín Sequeira, Sebastián Raffo y Camilo Melani, amigos que estuvieron siempre presentes.

A la Universidad de Buenos Aires y en particular al Departamento de Computación por brindarme una formación de excelencia y gratuita.

Este trabajo fue financiado mediante una beca de CONICET y una beca de la Universidad de Buenos Aires. También agradezco a la CONAE por brindarme gentilmente las imágenes con las cuales realicé mi investigación.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The exponential growth of data acquired by satellite sensors and the high cost of storing and transmitting these huge data volumes, present a challenge as to the design of compression techniques, which must be lossless for storage at ground stations, on account of ulterior processing requirements.

Compression schemes operate by reducing statistical redundancies present in the data; reducing spectral (interband) correlations, as well as spatial correlations, leading to higher compression ratios.

In this thesis lossless multiband image compression is addressed. Two types of images are used on which tests are performed: multispectral and hyperspectral. The former are captured by the LANDSAT satellite, one of the oldest missions sent out to space capable of capturing multiband images. The latter are obtained by the AVIRIS project, which has a more advanced and evolved tecnology in its sensors with respect to LANDSAT, since it provides images with more resolution.

This thesis is organized in three main chapters. In Chapter 2, in order to get familiar with the type of images that will be used throughout this thesis, a review of satellite images is presented and the two main satellites or missions are described. Depending on the capabilities of the sensors on the satellites, they are able to capture images with different characteristics.

These aspects such as the types of resolutions have a great influence on what methods or techniques are more suitable to compress the images.

In Chapter 4 we focus on multispectral images captured by the Landsat mission. A new lossless compression method based on prediction of wavelet coefficients taking into account classes in the images is presented. Two approaches are taken: offline and online. The former aims at speeding up the compression process while the latter takes advantage of the available data as the compressor is running. For each approach, different ways of establishing classes and band orderings are used.

In Chapter 5, hyperspectral lossless compression techniques are presented. Well-known compression methods are briefly mentioned and the popular lookup table-based methods LUT and LAIS-LUT are described. Then our prediction-based compression scheme is presented, which uses lookup tables as well as classification so as to make prediciton more accurate.

Chapter 2

Satellites and images

The term remote sensing refers to methods that employ electromagnetic energy, such as light, heat, and radio waves, as the means of detecting and measuring target characteristics. This electromagnetic energy refers to all energy that moves with the velocity of light in a harmonic wave pattern. The waves can be described in terms of their velocity (which is the speed of light when waves move through a vacuum), wavelength (distance from any point of one cycle to the same position of the next one), and frequency (number of wave crests passing a given point in a specified period of time; unlike velocity and wavelength, which change as electromagnetic energy is propagated through media of different densities, frequency remains constant and is therefore a more fundamental property).

When electromagnetic energy encounters matter, many of the energy properties are changed, and the science of remote sensing is the one that detects and records these changes by acquiring images that are interpreted and used to determine the characteristics of the matter.

The continuum of energy that ranges from meters to nanometers in wavelengths is called the electromagnetic spectrum (ES). It is divided into regions depending on the wavelength. For example, shorter wavelengths measured in fractions of nanometers belong to the gamma region while the radio region spans wavelengths on the order of meters (see Figure 2.1).

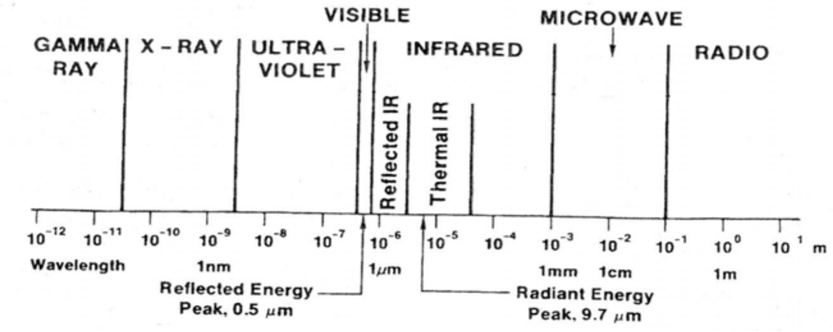


Figure 2.1: Regions in the electromagnetic spectrum.

In this thesis only passive remote sensing systems are considered. These systems record energy that naturally radiates or reflects from an object as opposed to active systems which supply their own source of energy directing it at objects and measuring how much of this energy is returned (for example, radar systems). Regular photographs are examples of passive systems, which record wavelengths from 0.3 to 0.9 μm that correspond to the visible range of the ES. Passive remote sensing systems are capable of capturing other regions of the ES as well; this makes them a versatile tool for detection of a wide variety of objects, among other applications. Because some regions in the spectrum are absorbed by the atmosphere (such as gamma rays, X-rays and most of the ultraviolet region) they are not used for remote sensing.

Images captured by pasive remote sensing systems can be described in terms of many characteristics. Attributes and features of each of these systems determine the characteristics of the images they acquire. All the images captured by the satellites or systems considered in this thesis are *multiband*. That is, for an image taken of a certain location, several bands are obtained. Each of these bands corresponds to the response of a certain range in the ES. For example, color photographic images are represented as three bidimensional arrays (or matrices), each array representing a different band of color: red, green or blue. In the same way, a remote sensor system could be able to capture three bands in the visible region (0.4 to 0.7 μm), each band corresponding to the appropiate color, in order to build a photograph as if it was taken with a regular camera. This multiband concept leads to one important characteristic of images: *spectral resolution*. Sensors with images captured at a higher spectral resolution are able to discriminate finer ranges of the ES and therefore more bands can be obtained in a given range of the spectrum. In this thesis two type of images are used: multispectral (formed by 7 bands) and hyperspectral (formed by over 200 bands covering wavelengths from the visible region to the infrared region).

Another important characteristic of satellite images is related to their scale, that is, the relation between two points on an image and their corresponding distance on the ground. This aspect, the *spatial resolution*, is important in order to determine what portion of the ground a pixel of an image represents. For instance, some applications require high resolution images in order to detect specific objects on the ground, whereas other applications that only need to segment types of crops on the ground, lower spatial resolutions are enough.

Remote sensing systems are able to detect different levels of the intensity of electromagnetic radiation that an object reflects, emits, or scatters at a particular wavelength (i.e., band of the image). The magnitude of this response is directly proportional to the brightness of that object in the image. Usually, each pixel is represented as an integer value where higher values indicate a higher response, and the range of possible values is assigned a grayscale from black (lower values) to white (higher values). This characteristic of images is connected with another type of resolution: *radiometric*. This resolution refers to the number of digital levels used to express the data collected by the sensor. Generally, it is expressed as bits per pixel.

There are other image characteristics but they are more related with their interpretation, which is out of the scope of this thesis. The variation in any of the aforementioned characteristics of images has a great impact on compression since methods developed for images of particular resolutions may not work when they are acquired with different resolutions. See [1, 2] for more details of the underlying physical principles of sensors, image interpretation and satellite missions.

2.1 LANDSAT images

The Landsat ('land satellite') program [3] was designed in the 1960s and aimed at observing broad-scale land areas of the Earth. There have been 7 missions launched since july 1972 and still 2 of them remain operational [4]. The first 3 missions (1972, 1975 and 1978) are commonly regarded as the *first Landsat generation* because of their shared characteristics (altitude, number of orbits, sensors, etc).

The imaging system on satellites consists of spacecraft-borne sensors, each designed for specific purposes. These are:

- **RBV** (Return Beam Vidicom): it generated high-resolution television-like images, providing 3 spectral channels (green, red and near infrared) each obtained by a separate camera that captures the corresponding segment of the spectrum. One of the intended applications was photogrammetry. This sensor was present in the first Landsat generation and dropped later because of their limited value images and technical difficulties.

- **MSS** (Multispectral Scanner Subsystem): this sensor was designed to provide multispectral data (by the use of a spectometer that disperses the incoming energy into a spectrum) regardless of the positional accuracy. It captured four spectral bands: green (0.5-0.6 μm), red (0.6-0.7 μm), and two near infrared (0.7-0.8 μm and 0.8-1.1 μm); the Landsat 3 MSS included an additional band in the far infrared from 10.4 to

12.6 μm. Although the MSS has been replaced in the last mission, it is significant because of the concepts it introduced and were later used in the TM system.

- **TM** (Thematic Mapper): it is an upgraded version of the MSS, which had an improved geometric fidelity and more precise definition of spectral regions, among other enhancements. TM captures 7 spectral regions and this sensor was part of Landsat missions 4 and 5. An enhanced version of the TM (ETM) was on the Landsat 6 mission in 1993 but it was lost at launch. In April 1999 the Enhanced Thematic Mapper Plus (ETM+) sensor was placed in orbit with the successful Landsat 7 [5] mission. It extended the capabitilies of the previous ETM by a better use of calibration and efficiency of data transmission but maintaining the spectral definitions and resolutions of the former sensor version.

The Landsat program is one of the pioneer observational programs and it was useful as an introduction to similar land observation satellites operated by other organizations. For the purpose -and at the time- it was designed the resolutions provided were moderate. In this thesis, Landsat 5 and 7 images are used. The spectral resolution provides for three bands in the visible region, and four in the infrared region (see Figure 2.1). Each pixel is allocated 1 byte yielding a radiometric resolution of 256 different digital values per pixel. Spatial resolution is about 30 meters.

2.2 Hyperspectral images

These images have a main distinction with respect to multispectral images: they provide for a higher spectral resolution. Now, 224 bands are available each with a spectral bandwidth of 10 nm in the region of 0.4 to 2.5 μm. Images captured by the scanner system called AVIRIS (Airborne Vis-

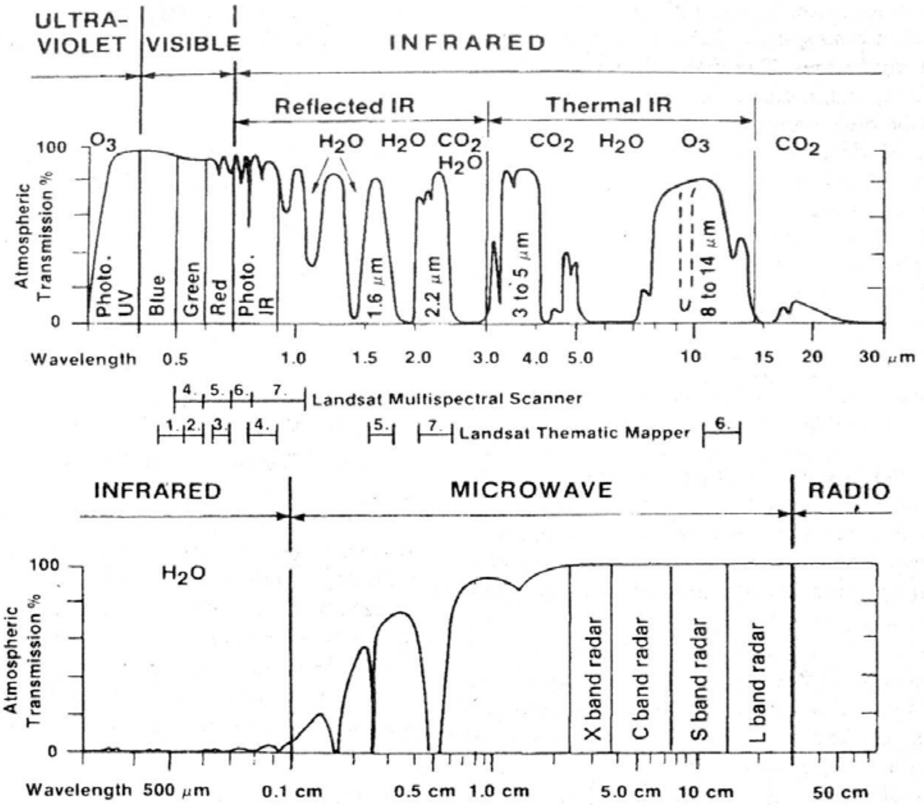


Figure 2.2: Expanded diagrams of visible and infrared regions (upper) and microwave regions (lower). Gases responsible for atmospheric absorption are indicated. Wavelengths in visible and infrared regions that are recorded by indicated Landsat MSS and TM bands are shown (middle).

ible/Infrared Imaging Spectometer) [6] are used in this thesis. It was developed by JPL (Jet Propulsion Laboratory) and is carried in a NASA U-2 aircraft at an altitude of 20 km. These images also have a higher spatial resolution than multispectral LANDSAT images. For AVIRIS images each pixel represents 20 m, and is allocated a 2 byte signed integer (which almost doubles the radiometric resolution of LANDSAT images).

Chapter 3

General introduction to compression techniques

Nowadays, media storage systems have increased their capacity cosiderably. Hundreds of gigabytes are available in small and portable devices at affordable costs. So the following question arises: why do we need compression? Well, compression is used in most of the data these days, for transmission or archiving. Take for instance a media library with thousands of music or video files. In order to digitally represent 1 second of video without compression, more than 20 megabytes, or 160 megabits, are needed using the CCIR 601 format (imagine the popular HD format). Two minutes of uncompressed CD-quality audio with 44100 samples per second and 16 bits per sample, require more than 84 million bits (10 megabytes). A lot of current file formats make use of compression: mp3/acc/ogg (audio), mpeg/mov/div-x (video), jpeg/png/tiff (image), pdf/doc/odt (documents) and many more. Without compression all the massive available storage would be worthless. Also, in spite of the fact that broadband internet access is becoming more and more popular, transmission of video or audio would not be feasible without compression.

The classical scheme for compressing data includes three sequential steps:

transformation, quantization and encoding. In some compressors these steps do not overlap, but somtimes it is hard to tell in which step of the process we are. During transformation, the original data to be compressed is changed so that its new representation is more suitable for compression. This may be thought as a change of basis. For instance, some input data are more compressible when expressed in a frequency domain. As another example, assume we have a large sequence of numbers $x_0, \ldots, x_n$ where $x_{i+1} := x_i + 1$. If the original sequence is expressed as differences $x_0, x_1 - x_0, \ldots, x_{i+1} - x_i, \ldots$ this new sequence is more compressible than the original one since less bits per symbol are needed to code it. In turn, given the 'compressed' sequence and the decompression algorithm, we are able to recover the original sequence without loss of information. This is called *lossless compression*. On the other hand, if an approximation of the original sequence is obtained after the decompression process, we are in presence of *lossy compression*.

During the second step is where the loss of information is produced. Quantization aims at reducing the number of symbols to code. For instance, if during the transformation step a source of integer numbers is transformed into a source of real numbers (which might potentially be all different), these transformed symbols could be less compressible. We could use a simple quantization function such as rounding each real number to its nearest integer so as to reduce the number of symbols to code. Even though data that is possibly more compressible is obtained, the original data cannot be recovered at the decompression process. Usually the quantization process is quite simple, but its design has a great impact on the amount of compression obtained and loss incurred in a lossy compression scheme. In lossless compression, this quantization step is omitted.

Finally, the coding step. After the symbols are transformed or quantized they are assigned a codeword. These codewords have their own alphabet. The Morse code is a classical example: this is a binary code whose codewords are formed by '$\cdot$' and '$-$' (dots and dashes is the binary alphabet for the codewords). It was designed taking into account the statistics of the english dictionary. It is an efficient code, since smaller codewords (i.e., shorter sequences of dots and dashes) are assigned to most frequent letters. In english words, the letter e is the one that most frequently occurs, while letter z is not that frequent. So, in Morse code, the shorter codeword formed by a dot '$\cdot$' is assigned to letter e, while a longer code '$- - \cdot\cdot$' is used to code z. Huffman code is another example, where a binary tree is built according to the statistics of the symbols. The final tree with the symbols on their leaves is used for encoding. The path to a leaf (starting from the root) indicates the codeword for the symbol on that leaf: each time a left branch is used, a 0 is added to the codeword and, in the same way, a 1 is added each time a right branch is taken on the path to the leaf. This tree is built in such a way that most frequent symbols end up closer to the root of the tree and therefore shorter codewords are used. Generally, binary codestreams contain codewords that are sequences of 0's and 1's which is the natural language of computers. Coding theory is vast and plenty of codes are designed in order to fit certain purposes and efficiently code particular sources. See [7] for more details.

3.0.1 Entropy

When data is compressed, we aim at creating a shorter representation for the data. For that, redundancies must be eliminated. There are several approaches for doing so and they depend on the type of data. First, a measure for determining how much is the cost to code a source of data is needed. In information theory, the information of an event E is determined by its probability $P(E)$: the more probable an event is, the less information it provides. When E takes place we say that $I(E) = \log(1/P(E))$ units of information have been received. If 2 is chosen as the basis for the logarithm, we refer to *bits* of information. Symbols to be coded belonging to a certain alphabet may be described mathematically as a discrete source of information $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ that emits symbols s_i with a fixed probability $P(s_i)$, $i = 1, \ldots, n$. The most simple source assumes that the emitted symbols are statistically independent. As stated before, the presence of a symbol s_i amounts to $I(s_i) = \log(1/P(s_i))$ bits of information and we may estimate the average information per symbol associated with the source as $\sum_{\mathcal{S}} I(s_i) \cdot P(s_i)$ and this is the entropy of the source, defined as

$$H(\mathcal{S}) \triangleq \sum_{\mathcal{S}} P(s_i) \cdot \log \frac{1}{P(s_i)}$$

Shannon [8] showed that the entropy is a measure of the average number of binary symbols needed to code the symbols emitted by the source $\mathcal{S}$. So, the entropy is a good measure to estimate how many bits are required to compress any source.

It was mentioned that in the coding process, a codestream is generated by assigning codewords to symbols. The average length of a code may be computed as

$$L = \sum_{\mathcal{S}} P(S_i) \cdot \ell_i$$

where ℓ_i indicates the length of the codeword assigned to symbol s_i. As an example, two different codes are presented in Table 3.1 for the source $\mathcal{S} = \{s_1, s_2, s_3, s_4\}$ whose symbols are emitted by the source with probabilities $P(s_1) = 1/2, P(s_1) = 1/4, P(s_1) = 1/8, P(s_1) = 1/8$ (or analogously, we have a sequence of symbols whose probabilities are the ones mentioned). It can be observed that CODE 1 is a fixed-length code, since codewords have the same length for every symbol ($\ell_i = 2$, for $1 \leq i \leq 4$). On the contrary, CODE 2 is a variable-length code.

The entropy of the source is $H(\mathcal{S}) = 1.75$. Since CODE 1 is fixed-length, it clearly has an average length of 2, while CODE 2 has a shorter average length and it is equal to the entropy. By a closer observation, CODE 2 is

CODE 1		CODE 2	
Symbol	Codeword	Symbol	Codeword
s_1	00	s_1	1
s_2	01	s_2	01
s_3	10	s_3	001
s_4	11	s_4	000

Table 3.1: Two different binary codes for a source of 4 symbols.

more efficient than CODE 1: it assigns shorter codewords to symbols that have higher probability (or have more repetition in a sequence of symbols).

The first theorem of Shannon states that the following inequality holds for any code that is uniquely decodable:

$$L \geq H(\mathcal{S})$$

3.1 Arithmetic encoding

The Huffman algorithm codes symbols with an average number of bits that is generally quite close (or equal) to the entropy of the source. Nonetheless, there are examples where Huffman fails to produce a code with such characteristics. Such is the case when the source has few different symbols and the probabilities are skewed. One solution to this is source extension. This approach groups symbols to form sequences and a codeword is generated for each possible sequence (i.e., sequences are considered as the new symbols). This approach has a clear drawback: as the length of sequences become larger, the number of possible sequences grows considerably; building the Huffman tree might be significantly time consuming.

The arithmetic coder follows a similar approach. Suppose a sequence of length m out of a source of n different symbols is to be coded ($m \gg n$). In order to distinguish among all the possible m^n sequences, the arithmetic coder assigns a different identifier to each different sequence. This identifier is some real number in the $[0,1)$ interval and will correspond to the codeword. The advantage of the arithmetic coder over Huffman, is that this identifier is built as each symbol of the sequence is coded and there is no need to generate all the possible m^n identifiers. So any source of symbols is coded with a real number (usually binary, although some integer implementations are available) in the range $[0,1)$ and the longer the sequence, the higher the precision needed to represent such number.

It can be shown that the longer the sequence to code, the closer the average length of the code (generated by the arithmetic coder) gets to the entropy.

3.1.1 Statistics update

The classical implementation of the arithmetic coder assumes that probabilities of the source being coded are known in advance. This means that we know that the symbols to be compressed are a realization of a source whose distribution is known to the coder. When dealing with data compression, the statistics of the source are rarely known in advance. That is why a first pass throughout the file or image may be needed to get their distribution, before the second pass where the actual compression process takes place. This is usually a disadvantage when demanding complexity requirements are imposed. This 'static' variant of how the coder handles statistics usually adds some overhead data to the codestream since the decoder must use the same statistics in order to recover the compressed source correctly. This overhead data may be avoided if both, coder and decoder, agree on the same statistics in advance. But if the source to be coded does not have that previously agreed distribution, the compression rates achieved may be far from the entropy.

The other variant is the 'dynamic' statistics update, where probabilities do not remain fixed during the whole process of compression. This is a more flexible scheme where no overhead data needs to be sent in the codestream. Coder and decoder just need to agree on any initial distribution –usually all symbols are equiprobable. Once the compression/decompression process starts, each time a symbol is coded/decoded, the probabilities are updated so as to reflect the distribution of the symbols already coded. Therefore, coder and decoder always share the same statistics as long they update the probabilites in the same manner.

The dynamic version is the usual implementation for the arithmetic coder in order to avoid scanning the source twice so as to obtain the distribution. Nonetheless, this implementation may be affected by the order in which symbols are coded. For example, suppose two files are to be coded, having symbols $\{s_1, \ldots, s_n\}$ with the same distribution. In the first file the symbols are ordered in such a way that after a small portion of this file has been coded, the statistics of the symbols read so far are similar to the statistics of the whole file and, as the compression process goes on, these statistics do not change too much. In the second file, on the other hand, symbols are placed in order: the first symbols are all s_1, then come all symbols s_2 and so on. It is clear that for this second file that, as the compression process ocurrs, the statistics estimated never reflect the distribution of the whole file until the last symbol is coded. Therefore, even though these two files have the same distribution, different compression rates may be achieved. This would not have happened with the static variant.

3.1.2 Coding with contexts

So far we have assumed that we have a model of the source represented by a set of probabilities, and we make use of this model every time a symbol needs to be coded. We also have assumed that this model has no memory: new symbols to be coded are supposed to follow the model but previously coded symbols are not taken into account. Consider the case of text compression. We could count the appearences of characters in words from the spanish dictionary and estimate a model for them. Now suppose the character *u* from word *que* is to be coded. Our current approach would code this character according to its probability but without taking into account that *q* is the previous character. Instead of using this memoryless model we could use a single-character context, that is, a probability model given *q* (or equivalently, a first-order Markov model). In spanish, if a *q* letter is observed it is highly probable that a letter *u* comes afterwards. This model would be more accurate instead of just considering the probability of letter *u* in the alphabet regardless of the preceding letter.

A similar concept is used when coding images by considering a neighbourhood of already coded pixels as a context. CALIC (Context Adaptive Lossless Image Compression) [9] is a good example of a context-based image coder. In CALIC, each pixel is predicted with a gradient-adjusted predictor. The bias in the prediction error is corrected with context information which takes into account spatial texture patterns and the energy of prediction errors. Then, corrected prediction errors are entropy coded using error energy (formed by a linear combination of the magnitude of gradients and previous prediction errors) as contexts.

By embedding contexts into the arithmetic coder, separate statistics may be used depending in which context the symbol to be coded occurs.

Chapter 4

Multispectral image compression

As introduced in the first chapter, one of the goals of this thesis is the design of a lossless compressor for multispectral images. In this chapter this compressor is described. First, the most important methods and techniques applied to compress multispectral images are mentioned. Then, the mathematical tool on which the proposed compressor is based is reviewed: the discrete wavelet transform (DWT). In the following sections several improvements are presented, each of which successively increases the compression rates achieved. They include classification of pixels and wavelet coefficients, ordering of bands, etc.

When the properties of multispectral images were briefly explained in Chapter 2, it was mentioned that thermal bands, which capture a wide range of the electromagnetic spectrum, have different characteristics compared to the other bands. That is why a different approach in the proposed compressor is taken for these bands; it is presented in Section 4.4. Finally, nonlinear prediction using neural networks applied on wavelet coefficients is described in Section 4.3.4.

4.1 State of the art for lossless compression of multispectral images

All image compression techniques operate by reducing statistical redundancies present in an image. To achieve this, standard single image compressors apply a wavelet transform (see Section 4.2). By representing an image as the sum of its details at different resolutions and orientations (plus a coarse approximation), a wavelet transform substantially reduces spatial correlations. Detail subbands have histograms that are peaked and centered at zero: their entropies are small. This is a remarkable property of the wavelet transform, and explains why wavelets are used in state-of-the-art image compressors. Traditional wavelets produce real coefficients. For lossless compression, wavelets that map integers into integers have been designed [10]. These transforms are reversible when the values of the original image are integers, as is the case. They are used in SPIHT [11], and JPEG 2000 [12, 13, 14]. Both compressors encode the most significant bitplanes first; JPEG2000 also uses contextual information.

Another approach to reduce spatial correlations, is to apply prediction–based methods, which have long been used. For each pixel being encoded, a prediction for that pixel is performed, using a small neighborhood of the pixel, called the prediction context or prediction set. The differences between the predictions and the original pixels are then encoded with an entropy based coder. The more accurate the predictions, the smaller the differences, and the higher the compression ratio. In the final coding step, sometimes the coding context (another neighborhood) of the pixels is used: different coding contexts define different coders. In order to perfectly recover the information at the decompressor's side, both the prediction context and the coding context must consist of pixels that have already been coded. CALIC [9] performs a prediction based on gradients in the prediction set, uses texture and gradient information to correct the prediction bias, and applies an adaptive entropy coder with a local error energy estimator as a coding context. JPEG–LS [15] uses a nonlinear median edge prediction, and for coding it uses a large number of context classes based on vertical and horizontal gradients.

Sometimes predictions involve parameters that are calculated over a large set of pixels called the training set. This is the case when linear predictions are made, and weights for the prediction must be calculated. Then either the weights are written into the code, or they must be recalculated by the decoder: in the latter case the training set must also consist of already coded pixels. ALPC [16] classifies the pixels locally, according to their own prediction set, and performs a different linear prediction for each pixel; weights of the linear prediction are calculated over the pixels belonging to the same class. Slyz et al. [17] predict a pixel with the average of past pixels having the same vector–quantized context. By conditioning the prediction to the class of the prediction context, or coding the prediction differences conditioned to the context, the mentioned compressors try to capture variations in the local statistics of the image.

When considering multiband images, and especially multispectral images, correlations between bands can be considerable. Reducing spectral (interband) correlations as well as spatial correlations will lead to higher compression ratios. Some of the mentioned 2D lossless compressors have later been extended for cubic data, such as multispectral images, hyperspectral images and video [18, 19, 20, 21]. The first two perform a wavelet transform on the 3 dimensions of the data. The last two include pixels from other bands/frames into the prediction. Wang et al. proposed different intraband and interband predictors, plus an ordering of the bands, for multispectral images [22].

In this thesis, predictions are not made on the value of pixels. Neither a

3D wavelet transform is applied. Instead, a different approach is taken: a 2D wavelet transform is performed on blocks of each image band, and then both the interband and remaining intraband correlations are reduced via an affine prediction of the wavelet coefficients. Distributions of wavelet coefficients vary with their resolution level, their orientation, and the landscape, so that different predictions for each stack/block/subband/class are considered.

Predictions made upon the wavelet coefficients of an image have been proposed before. Linear predictions of wavelet detail coefficients using information of wavelet coefficients at the same or at different scales have been designed for single images [23, 24]. For compression of multispectral images, Benazza et al. apply a wavelet transform to each band; a linear prediction, which involves pixels from several previous bands, is built into the lifting scheme of the wavelet transform [25]. Yet none of the proposed methods take into account variations due to the landscape: the training set is the whole wavelet subband, even though a mixture of landscapes may be present in it; therefore the weights they calculate are not optimal.

4.2 The wavelet transform

It was mentioned that decorrelation of images is related to their compression since a sparse representation is obtained after the decorrelation process, which favors compression. The wavelet transform is a versatile tool widely used in multiple scenarios and in many applications. It is good at decorrelating images and has become part of the (not so) new standard for image compression: JPEG 2000 [12]. Since it provides for spatial localization –as opposed to the Discrete Cosine Transform used in basic JPEG [26], although this lack of spatial localization is dealt with the partition of the image into small blocks–, it allows the inclusion of features such as quality and spatial scalability, region of interest, lossless compression, etc.

4.2.1 The S+P wavelet transform

For our compressor we chose to apply the S+P transform [27], that maps integers into integers: when tested on several multispectral images, the entropy of the S+P transform was lower than the entropy of other integer wavelet transforms [28]. Briefly, the formulae of the S (similar to Haar) and S+P transforms are given for 1D signals.

Given the original signal $\{s_n\}$, $(n = 0 \ldots N-1$, N even), its S transform (one step) is composed of 2 sequences that are half its length: $\{c_n,\ d_n\}$, $(n = 0 \ldots \frac{N}{2}-1)$, where $\{c_n\}$ are the approximation (or "lowpass" coefficients), and $\{d_n\}$ are the detail or "highpass" coefficients:

$$c_n = \left\lfloor \frac{s_{2n} + s_{2n+1}}{2} \right\rfloor, \qquad d_n = s_{2n} - s_{2n+1}. \tag{4.1}$$

The inverse transform for S is calculated as

$$s_{2n} = c_n + \left\lfloor \frac{d_n + 1}{2} \right\rfloor, \qquad s_{2n+1} = s_{2n} - d_n, \tag{4.2}$$

for $n = 0 \ldots \frac{N}{2} - 1$. Observe that there is a truncated average in the approximation coefficients. The proof of the reversibility of this transform is based on the properties: $\forall x \in \mathbb{R}$, $\forall n \in \mathbb{Z}$,

$$\lfloor x \pm n \rfloor = \lfloor x \rfloor \pm n, \qquad -\lfloor n/2 \rfloor = \lfloor (-n+1)/2 \rfloor .$$

The S+P transform [27] applies the S transform and then operates a differential prediction on the detail coefficients, rounding the result and subtracting it from d_n to give d'_n.

Let $\{c_n,\ d_n\}$ be the S transform of the signal s_n. For $n = 1 \ldots \frac{N}{2} - 2$, calculate a prediction for the details

$$\widehat{d_n} = \frac{1}{4} \triangle c_n + \frac{3}{8} \triangle c_{n+1} - \frac{1}{4} d_{n+1}, \tag{4.3}$$

where $\triangle c_n = c_{n-1} - c_n$. First and last prediction values are

$$\widehat{d_0} = \frac{1}{4} \triangle c_1, \qquad \widehat{d}_{(\frac{N}{2}-1)} = \frac{1}{4} \triangle c_{(\frac{N}{2}-1)}. \tag{4.4}$$

Now obtain the new detail coefficients d'_n:

$$d'_n = d_n - \left\lfloor \widehat{d}_n + \frac{1}{2} \right\rfloor, \qquad n = 0 \ldots \frac{N}{2} - 1. \tag{4.5}$$

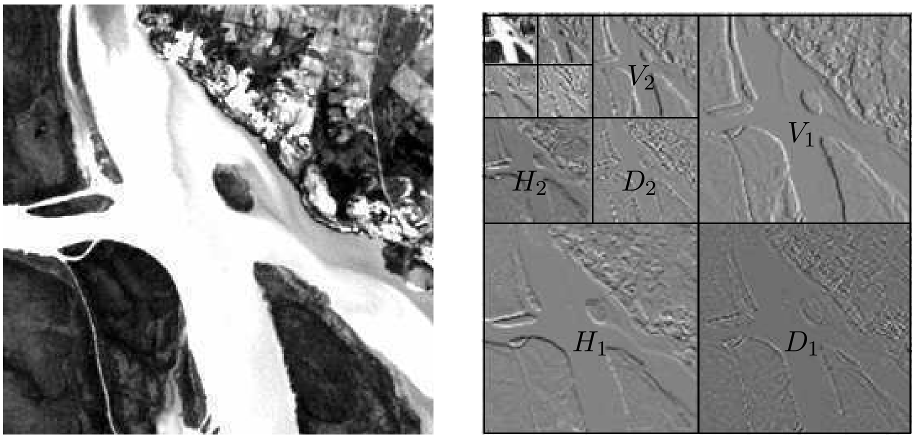
Then $\{c_n,\, d'_n\}$ is the S+P transform of s_n (one step). By substitution it may be shown that this is equivalent to filtering the data with 2 filters –a lowpass filter and a highpass one– and then downsampling the results. There is a rounding operation to give integer coefficients. Several steps are carried out by repeating the process on the lowpass filtered branch of the signal, so that the signal is split into its details at different scales and positions, plus a coarse approximation (of the signal).

The inverse S+P transform is calculated in reverse order, by applying equations (4.4), (4.3), and

$$d_n = d'_n + \left\lfloor \widehat{d}_n + \frac{1}{2} \right\rfloor, \qquad n = \frac{N}{2} - 1 \ldots 0. \tag{4.6}$$

When applied to an image, the same process is performed on the rows and resulting columns of the image. One step produces 4 subimages or subbands, one of which is an approximation of the image, and the 3 others capture the vertical, horizontal and diagonal details of the image. The process is repeated on the approximation subband. After several steps, the image is split into its details at different scales, orientations and positions, plus a coarse approximation.

In Fig. 4.1 (a) we have a 256×256 original image block. We show 3 steps of its S+P wavelet transform in Fig. 4.1 (b). The largest subbands V_1, H_1 and D_1 capture the vertical, horizontal and diagonal details, respectively, at the finest scale. Subbands V_2, H_2 and D_2 belong to the second fine scale. There is a coarse approximation LL of the image in the upper left corner. All the subbands have been rescaled independently; most of the coefficients are grey (close to zero) indicating that spatial correlation has been reduced.



(a) (b)

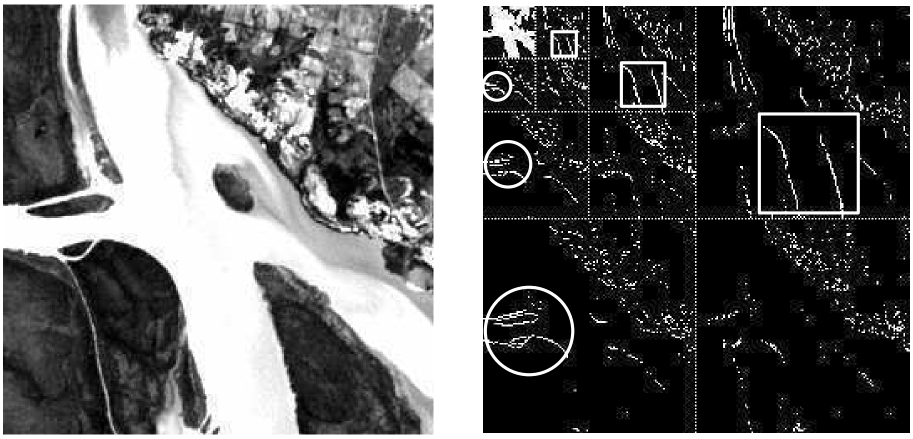
Figure 4.1: (a) A block of an image and (b) 3 steps of S+P wavelet transform of image in (a).

4.3 Prediction of wavelet coefficients

Wavelet transforms decorrelate images, eliminating spatial redundancy, i.e., the entropy of the transformed coefficients is much lower than that of the original image. Even so, similarities can be observed between the absolute values of the coefficients corresponding to the same spatial location at different scales – see the block of an original image in Figure 4.2 (a), and a thresholded version of its wavelet transform in Figure 4.2 (b), where we have the 9 detail subbands and 1 approximation subband of 3 steps of the S+P transform. Inside white circles and squares are observable similarities.

These similarities were used in [23] to state a model of statistical dependence among coefficients and it was applied to the lossy compression of photographic grayscale images. This model states that the dependencies are fixed inside each subband, and do not depend on the coefficients' actual value.

Based on the similarities observed in Figure 4.2 (b) and using the aforementioned underlying model, prediction may be performed upon wavelet coefficients. Since similarities are present when observing throughout scales

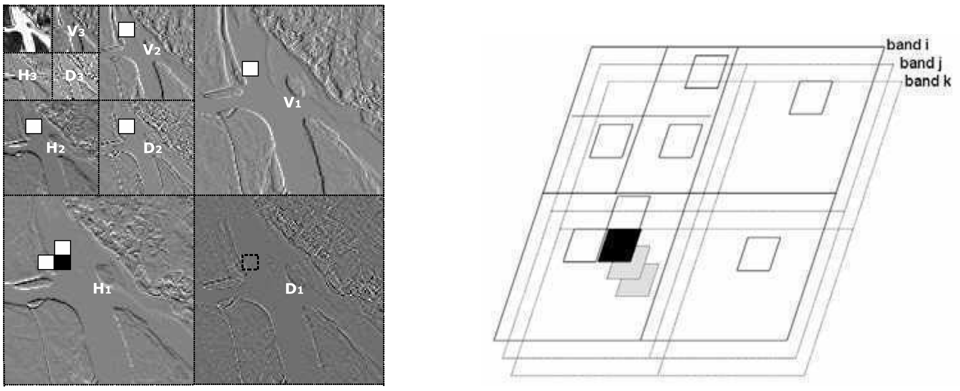


(a) Original block.

(b) Circles and squares indicates similarities between subbands.

Figure 4.2: (a) A block of an image and (b) its S+P transform (coefficients above a threshold in white)

but *at the same spatial location*, a wavelet coefficient belonging to a wavelet subband can be predicted with certain *candidate* coefficients. Consider Figure 4.3 as an example. In (a) the black coefficient in subband H_1 at the finest scale is considered for prediction; in white, candidate coefficients for prediction are shown. They may be located at other scales, orientations or adjacent to the coefficient to be predicted. These candidates may be combined -linearly, for instace- to generate a prediction for the coefficient in black. Then, the prediction error is entropy coded. Notice that candidates must be other coefficients that have already been coded so that the same prediction can be estimated when recovering the compressed image (decompression process). That is why the coefficient at the same spatial location in the finest D_1 subband is shown in dashed line; this means that this subband has not been traversed yet so this coefficient cannot be considered as a candidate for prediction. So it is assumed that each subband is predicted at a time, starting, as is the case in the figure, from coarser scales to finer scales (see next paragraph for more details). Since wavelet transforms from other bands are also available for prediction, the set of candidates is extended. If



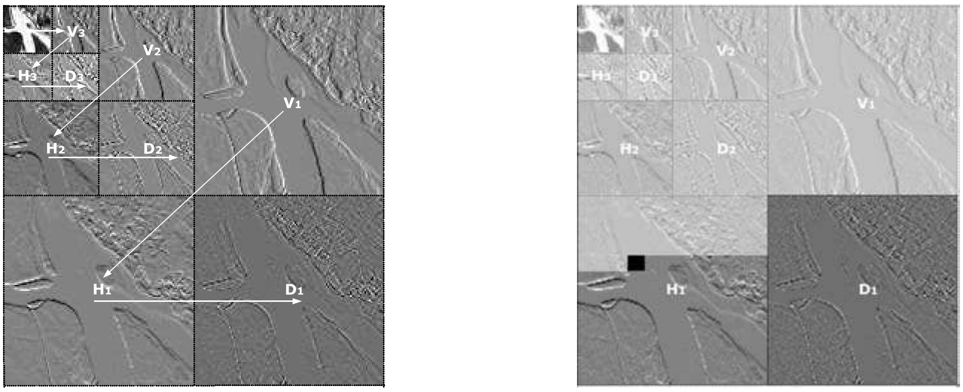
(a) Coefficient (dash line) not available for prediction, since H_1 subband has not been coded.

(b) In gray, candidates from already coded bands.

Figure 4.3: Current coefficient to be coded (black), with candidates in the same band (white) and in other bands (gray).

pixels between two bands are correlated, so are the coefficients of the wavelet transform. That is why wavelet coefficients at the same spatial location but on already coded bands belong to the set of candidates.This is shown in 4.3 (b).

Since the set of candidates only includes coefficients that were already coded, an ordering for encoding must be set. It has been shown in some works that statistics are not the same for every subband in the wavelet transform. For instance, when quantizing wavelet coefficients -either for achieving better compression rates or smaller errors-, global results improve when subbands are treated separately. Another example is the wavelet transform coding: if a compressor is restarted for every different wavelet subband, the compression rates achieved are higher than if all coefficientes are coded together. That is why coefficients in each subband will be treated and predicted separately. The order in which subbands are traversed is shown in Figure 4.4 (a). When considering one subband in particular, the coefficients within are predicted or sent in raster-scan order. In Figure 4.4 (b) the current wavelet coefficient to be encoded is shown in black. Also shown are the coefficients that have been encoded at this stage in light grey -these coefficients are available both to the encoder and to the decoder-, and in dark grey, the coefficients to be encoded.



(a) Ordering of wavelet subbands.

(b) Current coefficient (black) and encoded coefficients (light grey).

Figure 4.4: Traverse order.

As described earlier, Simmoncelli and Benazza performed predictions over wavelet coefficients in their works. The former applied it to grayscale 2D images while the latter extended the lifting scheme for multiband images. For the proposed compressor, the set of candidates is extended so as to consider coefficients from other bands as candidates as well. However, none of the aforementioned works take into account variations due to the landscape. In this work a further step is taken: the mixture of landscapes or clases in images is considered when performing predictions. When trying to predict coefficients, the best combination of candidates (that is, in a linear combination, the best weights) is estimated only taking into account the candidates that belong to the same class of the coefficient to be predicted. How classes are established and predictions computed, depend on the approach taken (offline or online; see Section 4.3.2 and 4.3.3).

4.3.1 General outline of the compressor

The compressor operates blockwise. For that, the image is split into blocks of 256 × 256 pixels. The group of blocks in different bands but at the same spatial location will be referred to as a *stack* of blocks. Each stack of blocks is coded independently by the compressor. The main advantage of splitting the image into blocks, is that it enables to reduce the number of operations. The wavelet transform has linear complexity in size, therefore, applying one step of the wavelet transform to each block of a band gives the same number of operations as applying one step of the wavelet transform to the whole band. However, to decorrelate a block of size 256 × 256, 5 steps of the wavelet transform are sufficient. And to decorrelate a whole image of size 6000 × 7000, many more wavelet steps should be performed. For this reason, applying the wavelet transform on a block basis reduces the number of operations.

A stack of blocks is fed at a time to the compresor, the blocks are scanned (one by one) in raster-scan order. Let N_b be the number of different bands in the image. *First subbands* (to be encoded) refer to the first 10 subbands (approximation coefficients and coarse detail): $\{LL,\ \{V_i,\ H_i,\ D_i\},\ i = 5, \ldots 3\}$. *Last subbands* (to be encoded) refer to 6 fine detail subbands: $\{V_2,\ H_2,\ D_2,\ V_1,\ H_1,\ D_1\}$ that are predicted. The steps performed for each stack are shown in Fig. 4.5.

As depicted in Fig. 4.5, though some wavelet coefficients are coded as is, most of the coefficients are predicted (the two finest scales represent the 93.75% of the whole transform). Also, no details were given about the classification process, since it depends on the approach taken: in the offline approach classes are determined before each stack is predicted and encoded, whereas classes are estimated anew for each block in a stack for the online approach.

```
foreach stack of blocks in an image do
    Initialization
        1 Calculate the S+P transform for each block.
        2 Select a band ordering.

    foreach block in a stack do
        Classification and Prediction
            1 Classify the pixels of the block.
            2 Compute predictions for the coefficients in the last
            subbands depending on the class they belong to.
            3 Calculate the prediction error.

        Coding
            1 Encode the coefficients in the first subbands separately,
            with an adaptive arithmetic coder.
            2 Encode the prediction errors for each of the last
            subbands separately, with an adaptive arithmetic coder,
            using the class as a context.
```

Figure 4.5: General algorithm for a stack of blocks.

4.3.2 Offline approach

In this approach [29], priority is given to lower the computational overload. For that, the following steps of the compressor are modified accordingly:

- **Class determination.** A supervised algorithm based on the spectral signature is used so as to determine which is the most numerous class in the stack. Based on observed types of landscape on available images, a group of classes is set. These are embbeded in the compressor and used each time a new stack is to be classified.
- **Band ordering.** Each of the classes also has a predefined ordering for bands. The ordering for each class is established offline taking into account the correlation between bands. Once a stack is set to belong to a certain class, the blocks are coded in the corresponding order.
- **Weight computation.** Weights for the linear combination are also computed in advance for each class, where sample stacks are used to train the best weights for each subband to be predicted.

Since speed is one of the goals for this approach, adjusting parameters (ie, which candidates, weights for prediction, best classification) to the data being compressed is avoided. Instead, fast algorithms based on offline training are chosen.

Supervised stack classification using spectral signature

If the reflectance of the earth is plotted in terms of the wavelength, different typical curves are obtained according to each landscape. These reflectance curves are called the spectral signature of the landscape [30, 5]. The reflectance curves for water, snow, vegetation and bare soil are shown in Figure 4.6. In order to determine the class of a stack, first each pixel considered as a vector in $\mathbb{R}^N$ (where N is the number of bands in the image) is classified according to its spectral signature.

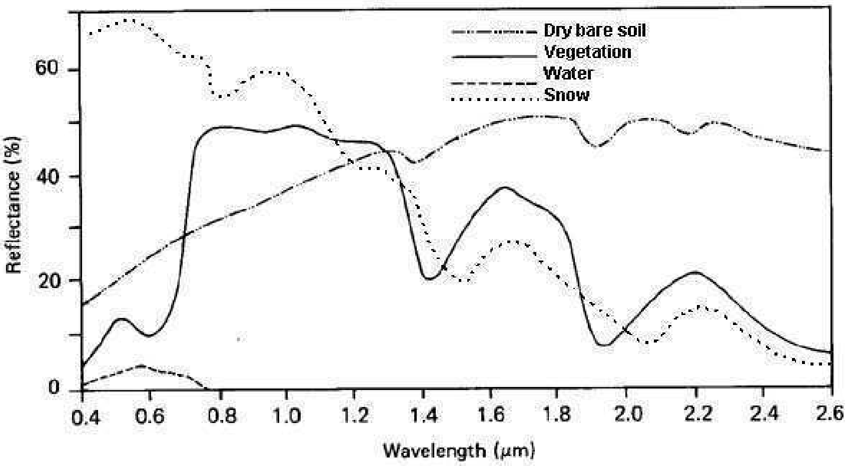


Figure 4.6: Reflectance curves for different types of landscapes.

Figure 4.7: A block to be classified.

We developed short ad hoc classification algorithms based on the shape of these reflectance curves.

For example, in the case of vegetation, it is noticed (Figure 4.6) that the reflectance curve for vegetation has a sharp increase at around wavelength 0.7 μm. Now, for each position (pixel) in a Landsat image we have 8 samples at the wavelengths given in Table 4.1. As a consequence of this increase, Band 3 < Band 4, so that, the algorithm classifies a pixel as vegetation if the well-known vegetation index Band 4 − Band 3 is positive. Similarly, a

Band	**Wavelength**
1	$0.45 - 0.52\ \mu m$
2	$0.52 - 0.60\ \mu m$
3	$0.63 - 0.69\ \mu m$
4	$0.76 - 0.90\ \mu m$
5	$1.55 - 1.75\ \mu m$
6.1,6.2	$10.4 - 12.5\ \mu m$
7	$2.08 - 2.35\ \mu m$

Table 4.1: Wavelength of each band in Landsat latest missions (thematic mapper sensor, see Figure 2.1).

pixel is classified as water if Band $4 -$ Band $3 < 0$ and the values for band 4 and band 7 are small. In case of dry bare soil, its reflectance curve is smooth and slighty upward. Although several factors such as humidity and chemical composition affect its spectral behaviour, the search of increasing and uniform values through Landsat bands turned out to be a good classification algorithm. Different kinds of soil without vegetation, as well as cities, fit into this category. If there are many classes inside a stack, *the class having the most pixels determines the classification of the whole stack.*

Figure 4.8: Pixels classified as water (in white).

For the block of Figure 4.7, the pixels classified as water are shown in Figure 4.8, the ones classified as vegetation are shown in Figure 4.9, and the ones classified as 'city' (bare soil) are shown in Figure 4.10. This block is classified as 'vegetation' since most of the pixels belong to that class.

Figure 4.9: Pixels classified as vegetation (white).

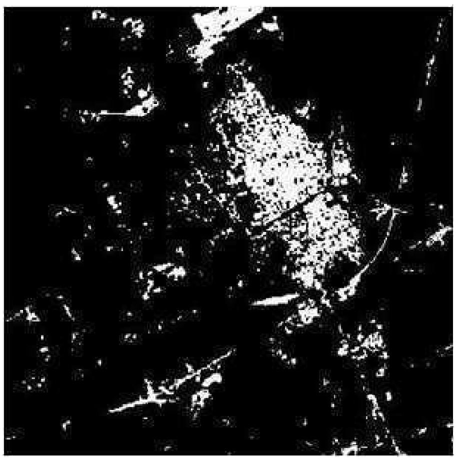


Figure 4.10: Pixels classified as city (in white).

Fast predictions based on offline training

Consider $s \in \{V_2, H_2, D_2, V_1, H_1, D_1\}$ as one of the subbands whose coefficients are to be predicted. Let S_s be the set of subbands whose coefficients may be chosen as candidates to predict coefficients in s, and let ζ_s be the number of elements in S_s (ie, the number of candidate subbands). In Figure 4.3 (a), $s = H_1$ and $S_s = \{V_1, H_1, H_2, D_2, V_2\}$. Since the subband s has 2 candidate coefficients, the total number of candidate coefficients is $\zeta_s + 1 = 6$.

Suppose a wavelet coefficient $= w^{(b)}_{i,j,s}$ is to be encoded, belonging to block b, at position (i, j) of detail subband s. Now let $\mathbf{w} \in \mathbb{R}^{\zeta_s+1}$ be the vector which contains all $\zeta_s + 1$ candidates for predicting coefficient $w^{(b)}_{i,j,s}$. The prediction for this coefficient is estimated by a linear combination of candidates. Let $\mathbf{a} \in \mathbb{R}^{\zeta_s+2}$ be the weights, with the $(\zeta_s + 2)$-th component being an offset. So, the prediction is

$$\widehat{w}^{(b)}_{i,j,s} = \mathbf{a}^t \begin{bmatrix} \mathbf{w} \\ 1 \end{bmatrix}$$

This version of the algorithm strives to decrease computational costs. Now, computing predictions as accurately as possible and using all available candidates becomes prohibited. By inspecting which candidates contribute most to predictions (ie, which weights are more significant) it is observed that the inclusion of all candidates does not provide for much bigger compression rates than using just a few. After several tests, it was decided to set to three the number of candidates to perform predictions since there was no significant improvement if more were added. Offline, a greedy algorithm is run to determine which are the best three, adding candidates one by one. This greedy strategy is applied on each subband and each class. This may be thought of as a training phase: for each of the classes determined according to the spectral signature several stacks of blocks are selected, which are used to train the best weigths for that class. The weights for the linear combination are calculated by least squares, computing the average over all stack of blocks of the corresponding class. Then, the weights estimated in the training phase, are incorporated into the compressor. This makes the compression process much faster, since once the class of the stack of blocks is established predictions may be computed rapidly using the proper weights.

It should be noticed that, if we have a mixture of classes in a stack of blocks, then inaccurate predictions may occur. Since the most numerous class determines the class of the whole stack, in the case of a stack of blocks composed of several classes or landscapes, the weights selected may not be the ones that yield best predictions (and therefore, bigger compression). Yet lossless compression is achieved.

Band ordering

With the purpose of maximizing the chance of obtaining predictions as accurate as possible, correlation between bands may be considered. In a first approach when offline training was used, it was observed that candidates that were chosen frequently to form the final set of three candidates and that belonged to a previously coded band, this band was highly correlated with the one to be predicted. So if bands are ordered in a way such that consecutive bands are highly correlated, when candidates from other bands are selected at the offline training process, predictions with these candidates would be more accurate than if they come from an uncorrelated band. Now, since a different set of candidates is chosen for each class of stack, a different ordering may be established for each class. All these orderings for each class are embedded in the compressor.

With all the previous considerations, the algorithm in Fig. 4.11 summarizes this offline approach.

Results

This approach of the compressor was tested on 4 Landsat 7 images from Argentina: Buenos Aires, Santa Cruz, San Luis and Mendoza. These images were used for the training phase, which include –for each of the classes

```
foreach stack of blocks $s$ in an image do
    Initialization
        1 Determine class of $s$
        2 Select a band ordering $b_{(1)}, b_{(2)}, \ldots, b_{(N_b)}$ depending on class of $s$.
    foreach block $b = b_{(1)}, b_{(2)}, \ldots, b_{(N_b)}$ in a stack $s$ do
        1 Calculate the S+P transform for block $b$.
        foreach of the last subbands $u = V_2, H_2, D_2, V_1, H_1, D_1$ do
            Prediction
                2 Compute predictions for the coefficients in $u$ using weights embbeded for the class of $s$.
                3 Calculate the prediction error.
        Coding
            1 Encode the coefficients in the first subbands separately, with an adaptive arithmetic coder.
            2 Encode the prediction errors for each of the last subbands separately, with an adaptive arithmetic coder, using the class as a context.
```

Figure 4.11: Offline approach of our general algorithm.

determined– the estimation of weights for the prediction of wavelet coefficients and the selection of a band ordering. In subsequent graphs we show how much is gained by the algorithms applied by our compressor for the image of San Luis. In Fig. 4.12 we have the entropy of each band of the original image (dotted) and the entropy of the wavelet S+P coefficients (solid); the gap between both curves indicates how much we have gained by applying the wavelet transform.

In Fig. 4.13 we have the entropy (bits per pixel) for each band of the

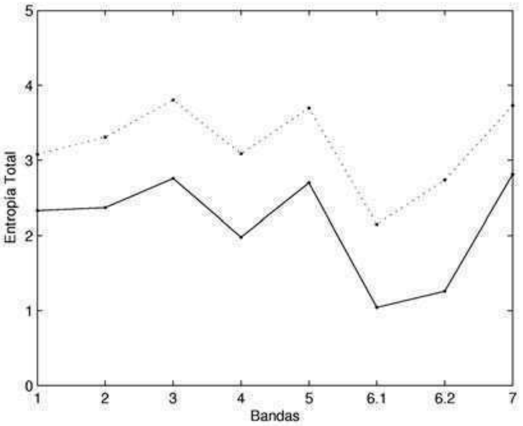


Figure 4.12: Entropy of the S+P transform (solid line) vs original image (dotted line).

S+P wavelet coefficients (dotted), versus our method based on the prediction of wavelet coefficients. The gap indicates how much we have gained by prediction.

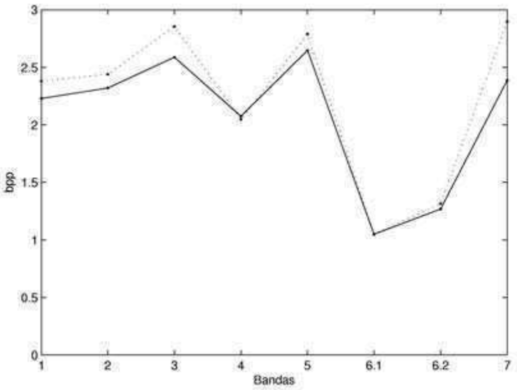


Figure 4.13: S+P transform(dotted) vs our method (solid) in bpp.

In Fig. 4.14 we have the performance of our method, in bpp, compared to the results of other lossless compressors: Winzip, LOCO-I, JPEG2000 and PNG (notice that these compressors do not benefit from spectral correlations since they are applied on each band separately). They are applied to the San Luis Landsat image and the different shades of the bars indicate the volume occupied by each band in the compressed image. We have the final results of the 5 mentioned lossless compressors applied to the mentioned 4 images of Argentina, in Table 4.2. Landsat 8-band images have 8 bpp. For the image of San Luis, each band has 7044 rows and 7476 columns. For San Luis, our method gives 2.07 bpp, that is, a saving of 5.93 bpp, and a compression rate of 3.86:1.

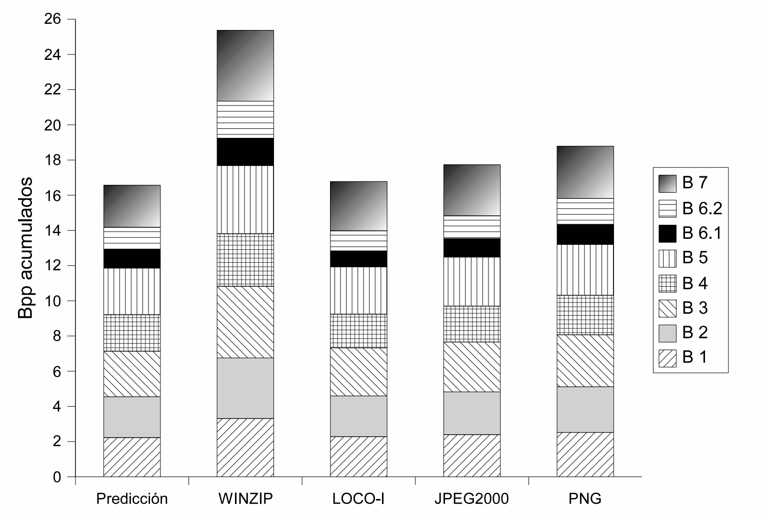


Figure 4.14: Results in accumulated bpp.

	PRED	**WINZIP**	**LOCO-I**	**JPEG2000**	**PNG**
Buenos As	104.76	161.05	106.88	113.53	121.67
Santa Cruz	90.37	139.94	92.43	98.54	105.61
San Luis	103.98	159.24	105.33	111.3	117.92
Mendoza	116.72	179.8	114.83	125.1	136.65

Table 4.2: Results in MB.

4.3.3 Online approach

All the actions that were taken before the compressor started running in the offline approach, now are performed as the compression process occurs [31]. Only the order in which the blocks in a stack are encoded is determined before the compression starts. The main differences are:

1. weights for the linear combination are estimated for each subband and class;
2. classes for pixels in a block are established based on the block that has been previously coded (in the same stack);

Suppose a wavelet coefficient $w = w^{(b)}_{i,j,s}$ is to be encoded, belonging to block b, at position (i, j) of detail subband s, where $s \in \{V_2, H_2, D_2, V_1, H_1, D_1\}$.

It is conjectured that pixels/ coefficients at the same position in 2 consecutive bands belong to the same class (see Section 4.3.3). The prediction for block b is based on the classes of block $b - 1$.

Let $w^{(b-1)}_{i,j,s}$ belong to class c. Suppose that $w^{(b-1)}_{i,j-1,s}$ and $w^{(b-1)}_{i-1,j,s}$ also belong to the same class. It is presumed that $w^{(b)}_{i,j,s}$, $w^{(b)}_{i-1,j,s}$ and $w^{(b)}_{i,j-1,s}$ also belong to class c. The prediction $\widehat{w}$ for w is calculated as an affine combination of the coefficient at the same position in the previous band, and the left and upper neighbors of w:

$$\widehat{w}^{(b)}_{i,j,s} = a^{(c)}_3 \; w^{(b-1)}_{i,j,s} + a^{(c)}_2 \; w^{(b)}_{i,j-1,s} + a^{(c)}_1 \; w^{(b)}_{i-1,j,s} + a^{(c)}_0 . \tag{4.7}$$

If $w^{(b-1)}_{i,j-1,s}$ does not belong to class c, its weight $a^{(c)}_2$ is set to zero since it is presumed that the left neighbor of w does not belong to class c. Similarly, if $w^{(b-1)}_{i-1,j,s}$ does not belong to class c, then $a^{(c)}_1$ is set to 0.

To calculate the weights $a^{(c)}_k$, $k = 0 \ldots 3$, the training set consists of all the coefficients in the same subband that belong to class c, and have their left and upper neighbors in the same class. They are calculated by least squares, thereby minimizing the Euclidian norm of the prediction error in the same class and subband. They are quantized to 10 bits.

The prediction is rounded and the prediction differences encoded

$$\Delta w_{i,j,s}^{(b)} = w_{i,j,s}^{(b)} - \left\lfloor \widehat{w}_{i,j,s}^{(b)} + 0.5 \right\rfloor . \tag{4.8}$$

At the decompressor side, prediction differences are decoded, and also the same prediction is performed, recovering w from

$$w_{i,j,s}^{(b)} = \Delta w_{i,j,s}^{(b)} + \left\lfloor \widehat{w}_{i,j,s}^{(b)} + 0.5 \right\rfloor . \tag{4.9}$$

Any error in the assumptions will result in less accurate predictions and less compression ratio; it will affect the decoder in no other way.

When processing the first block, no previous band is available: in this case prediction is intraband ($a_3^{(c)} = 0$). None of the classes are available either; therefore the weights are calculated over the whole subband. It should be noticed that both, the original block and their S+P transforms, must be kept in memory. Also, some overhead information must be sent to the decoder: block ordering is stored in the stack header and a total of 48 weights (4 parameters ($a_3^{(c)} \ldots a_0^{(c)}$) × 2 classes × 6 subbands) must be stored into the block header. For the first block, classification is skipped and, since no previous band is used for prediction, only 18 weights are stored into the block header. A block diagram for this approach is shown in Figure 4.15.

In the compression process there are various steps which are non-linear. These are: 1) in each stack of blocks the bands are ordered differently, 2) an offset is added to the linear prediction, 3) the prediction performed is different for each stack of blocks, depending on the band, subband and class, 4) there are rounding operations both at the S+P wavelet transform and at the prediction, This nonlinearity enables it to exploit complex dependencies.

In [20] Wu and Memon observe that correlation between 2 bands can vary spatially with the landscape; therefore it is counterproductive to use interband predictors in case of weak interband correlation. They propose switching between interband and intraband prediction according to the correlation in a local window. For the online version of the proposed compressor, the best weights for prediction are learned for each subband and each class of landscape: when the correlation between the bands is locally high, the calculated weight of the coefficient in the preceeding band will be high; when the correlation is low, the same weight will be comparatively small, so that the compressor also "switches" from interband to intraband prediction if the correlation is low.

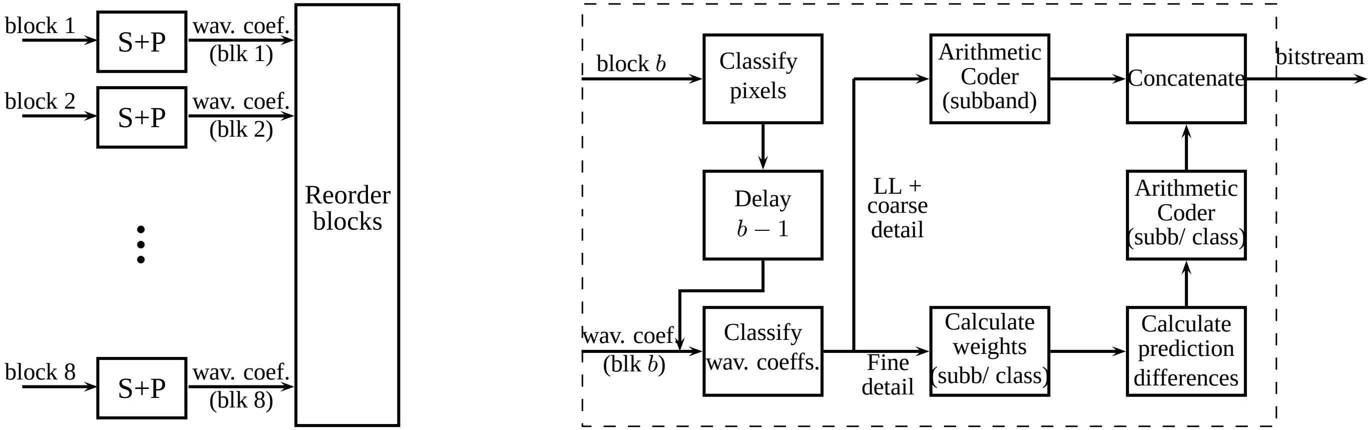


Figure 4.15: Compressor scheme. Left: Initial step for a stack. Right: core of the compressor for a particular block.

Unsupervised Classification

The former classification algorithm (see Section 4.3.2) was done on a stack basis. It assigned a landscape to each stack of blocks, and was based on the spectral signatures (i.e. typical behaviours of the reflectance in terms of the wavelength) of the different landcover types. Prediction weights (for each landscape, band and subband) were calculated offline on typical stacks of different landscapes, and stored into the compressor. One drawback was the impossibility to include weights for all possible kinds of landscapes. Another drawback was the inferior compression ratios obtained when a mixture of landscapes was present in a stack.

An online approach was also taken for classification. In order to avoid the problems mentioned in the last paragraph, now the classification is performed *i*) with an algorithm needing no previous knowledge on landscapes, and *ii*) on a pixel basis. A Lloyd–Max [32] quantization algorithm was used to determine the classes. Separating the pixel values into 2 classes converges in 2 or 3 iterations.

Classifying vectors (involving all the blocks in a stack), would require sending the template with the classes for reconstruction, a prohibitive overhead. Instead, pixels are classified inside one block. The classes and the weights (for each landscape and subband) are calculated anew for each block. According to a previously established order for band coding (see next Section), each time a new block needs to be coded, the Lloyd–Max algorithm is run on the previously coded block so as to determine the clases for this new block. Based on tests, there is no significant increase in the compression ratio when more than 2 classes are considered: accordingly, pixels in each block are separated into 2 classes.

For reconstruction it is necessary to have exactly the same classes at the decompressor side. Both the compressor and decompressor must determine the classes in an identical way: this can only be done if the classification

1	0	1	1	⋯
0	0	1	1	⋯
1	1	0	1	⋯
0	0	0	0	⋯
⋮				

0	1	⋯
1	0	⋯
⋮		

Figure 4.16: Left: Original template. Right: Second template.

involves data that is available to the decompressor. Therefore predictions are based on the classes of the block previous to the one being encoded. That is, assuming correlation between 2 consecutive bands to be significant, it is expected that pixels at the same position in 2 consecutive bands, generally belong to the same class.

The initial cut-off value determining 2 quantization intervals for the first block is fixed. This value and the centroids (one for each interval) evolve through Lloyd–Max iterations until they converge. The classes obtained are used to calculate weights for the second block. These are written into the code and related to the class through their index. The initial cut-off value for blocks 2 and onwards are set to the final cut-off value obtained in the previous block.

The pixels of the image block are separated into 2 classes turning out in a 256 × 256 binary template indicating the class of each pixel. This in turn induces a classification of the wavelet coefficients in the 6 predicted subbands. First, a second template is built that is a fourth (in size) of the original one, to indicate the classes in subbands V_1, H_1 and D_1. For this, the original template is divided in 2 × 2 submatrices, and the most numerous class in a submatrix is an entry in the second template – see Fig. 4.16. Similarly, a third template indicating the classes in subbands V_2, H_2 and D_2 is obtained. These templates are used to control that predictions are restricted to a class.

After separating the pixels of Fig. 4.2 (a) into 2 classes: land and water, wavelet coefficients have been classified –see Fig. 4.17.

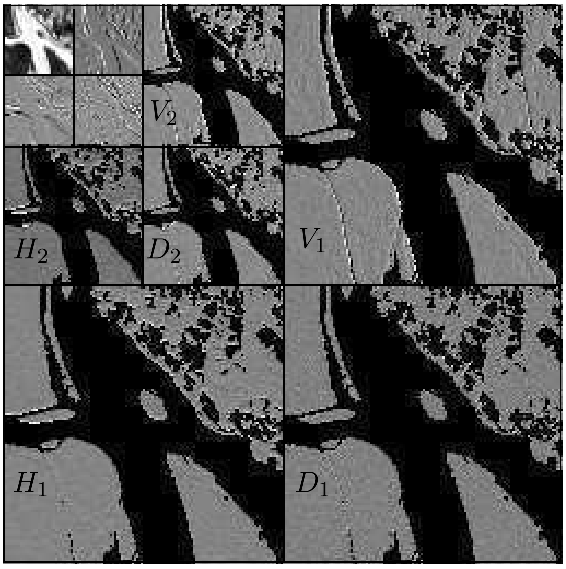


Figure 4.17: S+P wavelet transform of the original block (3 steps). In black: detail coefficients (of the 2 finest scales) classified as water.

Band Ordering

The order of the bands affects the accuracy of predictions, because coefficients from the previous band are included in the affine combination. It is therefore desirable to have an optimal band ordering, so that highly correlated bands are coded consecutively. This ordering may not be unique and must be sent to the decoder to recover all the bands in the right order [33]. Our compressor operates on one stack of blocks after the other, and determines a different band ordering for each stack. Originally this order was determined according to the correlation between blocks inside a stack. Since the wavelet transform decorrelates an image, correlations are lowered by the transform, and the optimal order may change. Our predictions are made on the wavelet coefficients: it was therefore natural to fix an order according to the correlations of the S+P transformed blocks (only considering the 6 subbands for which prediction is done). Doing so improved results [34].

Inside a stack of blocks, from now on the block in band b will be refered as block b. The mechanism is as follows: for each transformed block b, with a greedy algorithm an order O_b is found that starts with b, and at every step chooses the transformed block having the greater correlation with the previous one. From the N_b orders found, then $\widetilde{O}$ is selected, the one having the greatest sum of correlations. Formally, $W^{(b)} = \{w^{(b)}_{i,j,s}\}$ is the set of wavelet coefficients belonging to the block in band b, of subband $s \in \{V_2, H_2, D_2, V_1, H_1, D_1\}$. For $b = 1 \ldots N_b$, set $O_b(1) = b$ and $O_b(k+1) = \arg\max_{\ell \neq O_b(1), \ldots O_b(k)} |\rho(W^{(k)}, W^{(\ell)})|$. The final order is then $\widetilde{O} = O_{\widetilde{b}}$, where

$$\widetilde{b} = \arg \max_{1 \leq b \leq N_b} \sum_{k=1}^{N_b - 1} \left| \rho(W^{(O_b(k))}, W^{(O_b(k+1))}) \right| .$$

Results

Our compressor was tested on different Landsat 7 images of Argentina.[1] In Table 4.3 we give the entropy of each original image, averaged over all the bands (1st column) and compare our compressor LLWPred to different available lossless compressors. These are: 2D compressors PNG[35], JPEG-LS[15], SPIHT [11] and JPEG2000 [14], and 3D compressors SLSQ-OPT [36], D-JPEGLS (or differential JPEG-LS, in which the difference between 2 consecutive bands are coded with JPEG-LS), JPEG2000 Color (every 3 bands are set as color components in JPEG2000) and 3D-SPIHT[18]. We have also implemented a cubic S+P transform, and DEC3, a wavelet-based interband prediction with lifting [25]. Both were followed by arithmetic coding. Results are given in bits per pixel; the best result for each image is enhanced.

---

[1] Results listed in Table 4.3 correspond to the Landsat 7 ETM+ images (Path /Row - adquisition date (in yyyymmdd format)): 228/084-20011126; 225/084-20030127; 229/094-20020715; 221/074-20030523; 227/079-20020615; 229/092-20010423; 231/083-20010115; 232/084-20011021; 227/089-20030501; 230/087-20030506; 231/078-20030529; 226/098-20020710; 229/088-20030515. We thank CONAE for providing them.

Clearly our compressor outperforms the others. For each image tested, the final bitrate given by our compressor is around $\frac{2}{5}$ of the image's entropy. On average our compressor used 1.89 bpp to store the images. Since they originally used 8 bpp, this represents a compression ratio of 4.23 :1.

#	H	PNG	JPEG-LS	SPIHT	JPEG2000	SLSQ -OPT	D-JPEGLS	JPEG2000 Color	3D S+P	3D-SPIHT	DEC3	LLWPred
1	5.28	2.58	2.24	2.43	2.43	2.62	2.26	2.36	3.43	2.70	2.36	**2.18**
2	4.81	2.40	2.11	2.24	2.24	2.49	2.17	2.19	3.24	2.49	2.19	**2.02**
3	4.37	2.03	1.78	1.90	1.90	2.08	1.65	1.82	2.54	1.93	1.80	**1.66**
4	4.98	2.32	2.21	2.39	2.45	2.53	2.34	2.44	3.42	2.78	2.26	**2.20**
5	3.86	2.09	1.86	2.00	1.99	2.16	1.95	1.96	2.87	2.28	1.96	**1.84**
6	4.17	1.96	1.70	1.84	1.83	1.99	1.67	1.77	2.46	1.96	1.77	**1.66**
7	4.78	2.35	2.10	2.22	2.22	2.36	2.06	2.12	3.19	2.38	2.12	**1.93**
8	5.51	2.77	2.33	2.55	2.54	2.70	**2.16**	2.33	3.53	2.61	2.39	2.20
9	3.89	1.70	1.58	1.64	1.68	2.00	1.71	1.72	2.56	1.96	1.60	**1.53**
10	3.98	1.90	1.79	1.88	1.94	2.23	1.88	1.93	2.90	2.21	1.84	**1.74**
11	5.21	2.61	2.54	2.81	2.87	3.04	2.49	2.77	3.79	2.99	2.43	**2.41**
12	3.86	1.72	**1.58**	1.72	1.74	2.05	1.64	1.74	2.53	1.90	1.60	**1.58**
13	3.65	1.81	1.71	1.80	1.83	2.11	1.80	1.85	2.67	2.08	1.73	**1.66**
Avg.	4.49	2.17	1.96	2.11	2.13	2.34	1.98	2.08	3.01	2.33	2.00	**1.89**

Table 4.3: Bitrates obtained with different lossless compressors

4.3.4 Prediction with neural networks

In the previous sections linear dependencies among wavelet coefficients were exploited. Now, we turn to a tool that will help us to take advantage of possible nonlinear dependencies: neural networks. We chose a simple architecture based on a multilayer feedforward neural network, with nonlinear transfer functions associated to the hidden layer, and a linear transfer function associated to the output layer. A description of the network architecture and training algorithm are presented in the next section.

The compressor presented in this work acts in a similar way as before. The two finest scales (which represent the 93.75% of the coefficients) are predicted and the prediction error is entropy coded; the rest of the subbands are encoded as they are. For each of the predicted subbands a neural network is trained with an input (the candidates for each coefficient) and a desired output (every coefficient in the subband). Once the parameters are learned, the coefficients are fed into the network in order to compute the prediction (network output). This prediction is rounded (so as to ensure lossless compression) and the prediction error is coded with an arithmetic coder. Now, the selection of candidates is no longer made during the training phase. Based on observations, for a given coefficient, the coefficients on the left, on the right, and in the same location but in the previously coded band, were the most frequently selected. So now, only the neural network parameters are established during the training phase.

For the decompression process, the neural network parameters need to be known in order to be able to compute again the prediction and then add it to the decoded difference. This enables the exact recovery of the coefficients. On the other hand it adds some overhead information to the codestream: each neural network configuration used in each subband must be sent (i.e., weights for the neurons' links).

Because of the greater overhead that class conditioned prediction would entail, we omit the classification step when predictions are performed with neural networks. Although the overhead of the mask for each block and band is so large that this method is computationally prohibitive, we include the method in this thesis to show that neural networks are a powerful tool that may be used to generate nonlinear predictions, which will generally be more accurate than in the linear case.

Before training the neural networks, a band order is established for each stack of blocks, which ensures that highly correlated bands will be consecutive (see Section 4.3.3).

Neural network architecture

We use two-layer feed-forward neural networks as nonlinear coefficient predictors. The neural network architecture, shown in Fig. 4.18, has three input units associated with the candidate values, a hidden layer of nine neurons or nodes, and an output unit. Neurons in the hidden layer have full connections from the nodes in the previous (input) layer and the proceeding (output) one. Weight values are associated to each connection in the network. The output at a node is a function of the weighted sum of the connected nodes at the previous layer. In our case, the output of a hidden neuron is the result of applying a nonlinear transfer function f to the weighted sums of the inputs. This function $f : \mathbb{R} \to \mathbb{R}$ is

$$f(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}$$

The last (output) layer applies the identity function $f(x) = x$ to a weighted sum of the outputs of the hidden layer; i.e., the output of the neural network is a linear combination of the outputs from the hidden layer.

Once the architecture is defined, appropiate weights $w_{i,j}^{(k)}$ must be learned. For that, the network is trained aming at minimizing the cost function (defined as an average of the squares of errors). The usual approach consists

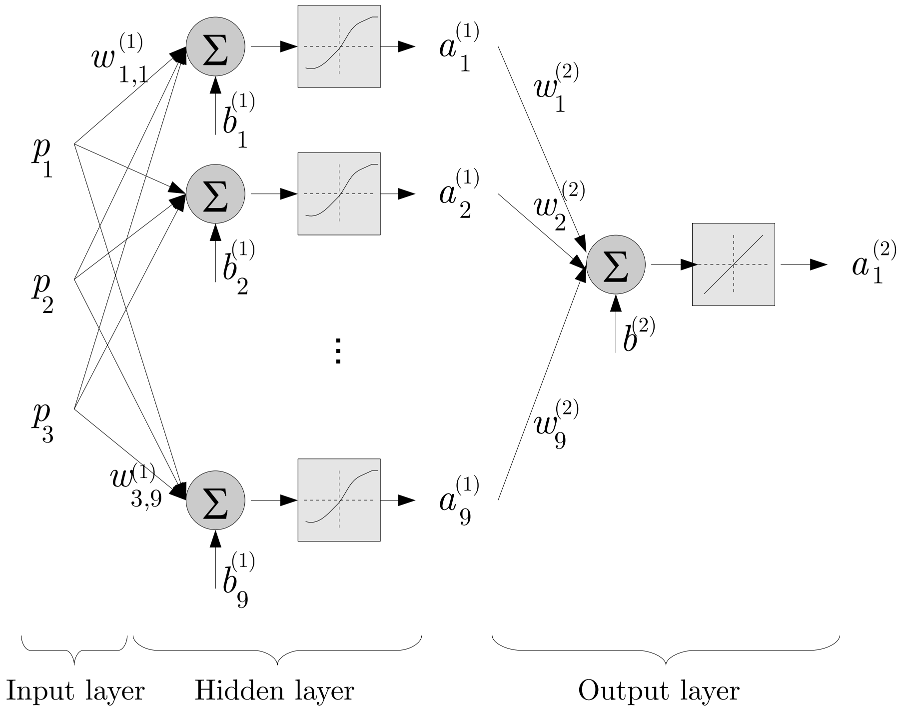


Figure 4.18: Architecture used in this work: p is an input value; w is a connection weight; b represents a bias value and a an unit output. The first layer has 9 neurons and an associated nonlinear function; the output layer performs a linear combination of the outputs of the previous layer.

of a stochastic back-propagation algorithm based on gradient descent. The algorithm gives a prescription for changing the weights w for each input pattern chosen in random order to learn a training set of input-output pairs [37] [38]. This strategy decreases the cost function (for small enough learning parameter) at each step, and lets successive steps adapt to the local gradient. Although the method allows wider exploration of the cost surface, it does not necessarily produce the fastest convergence. So, we use conjugate gradient algorithm for training, where a search is performed along conjugate directions and we get faster convergence for our experiments. In most of the conjugate gradient algorithms, the step size is adjusted at each iteration. A search is made along the conjugate gradient direction to determine the step size which minimizes the cost function along that line. We used a variation to avoid this time-consuming line search known as the scaled conjugate gradient algorithm[39].

Results

Tests on sample stacks of blocks belonging to Landsat 7 images are presented. The performance of the neural network is compared with the least squares approach. For that, instead of training a neural network, the prediction is computed by means of a linear combination whose weights are estimated via least squares. The bitrates achieved by different methods (including our method with neural networks and using linear prediction as in our previous works) are listed in Table 4.4, for one stack of blocks of size $256 \times 256 \times 8$. We trained neural networks with the architecture and algorithm depicted in the previous Section. For the finest scale, three neural networks (one for each orientation) were used and trained with 128×128 patterns; for the scale in a coarser level of detail less patterns are used: we have 64×64 coefficients to predict in each orientation.

In Table 4.5 we summarize results for some sample stack of blocks that belong to different areas of Argentina. As in the previous table, other lossless methods like LOCO-I, JPEG2000 and PNG are included. The nonlinear approach of the neural network method, lowers the bitrates achieved by the linear prediction of coefficients.

Block	**Neural Net**	**Linear Pred**	**JPEG 2000**	**LOCO-I**	**PNG**
1	2.63	2.64	2.76	2.63	3.04
2	2.46	2.47	2.78	2.61	3.04
3	2.83	2.85	3.11	2.95	3.32
4	3.36	3.37	3.80	3.67	3.88
5	3.18	3.20	3.81	3.65	3.88
6.1	1.28	1.28	1.33	1.11	1.43
6.2	1.49	1.49	1.57	1.39	1.73
7	3.17	3.18	3.47	3.30	3.61
Total	2.55	2.56	2.83	2.66	2.99

Table 4.4: Bitrates for each band of a sample stack of block of Chaco (Argentina). First column shows our results and second column corresponds to the linear prediction used in previous works.

Sample Stack	**Neural Network**	**Linear Pred**	**JPEG 2000**	**PNG**	**LOCO-I**
Chaco	2.550	2.560	2.830	2.991	2.661
Santa Cruz	2.406	2.418	2.691	2.889	2.525
San Luis	2.429	2.436	2.697	2.863	2.550

Table 4.5: Results for sample stacks of blocks of Landsat 7 images of Argentina.

4.4 Context-based edge prediction for thermal infrared band

The thermal bands in Landsat images are responses captured from the far infrared region ($10.4-12.5\mu m$ wavelengths) of the electromagnetic spectrum. This region is captured with a coarser spatial resolution compared to the resolution of the other bands[2]. Initially the MSS sensor captured far infrared in band 8 in the Landsat 3 program. Later, the TM (Thematic Mapper) was in charge of capturing this region (as well as their enhanced versions ETM and ETM+). The information in these thermal bands has several applications such as vegetation stress analysis, soil moisture discrimination, thermal mapping and plant heat stress, etc.

In Landsat 7, all ETM+ bands –excluding band 6– have an adjustable 'gain' setting, which allows the Mission Operations Center (MOC) to maximize the instrument's radiometric resolution for each band without saturating the detectors. Generally, acquisition of a scene in low gain allows a higher dynamic range with less risk of saturation over bright areas. Acquisition in high gain will provide greater radiometric resolution (sensitivity). Band 6 is acquired in two gain settings: low (6L) and high (6H), each provided in a separate band file. Band 6L provides an expanded dynamic range and lower radiometric resolution (sensitivity), with less saturation at high Digital Number (DN) values, while Band 6H has higher radiometric resolution (sensitivity) with a more restricted dynamic range (see Fig. 4.19).

A special algorithm was designed for the compression of band 6L [28]. The histogram of this band is very similar to that of band 6H. Consider a few histograms of a $256 \times 256 \times 8$ stack of blocks of the image of San Luis having coordinates $x = 2817$, $y = 1537$: in Fig. 4.20 we have the histograms

---

[2] ETM+ Band 6 is acquired at half the resolution (60-meter) of other bands (30-meters) although newer products are resampled to match the spatial resolution of other bands.

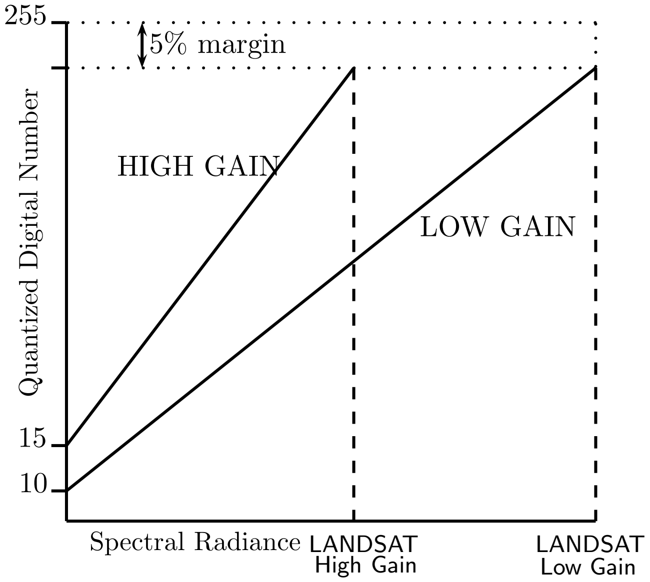


Figure 4.19: Design ETM+ reflective band high and low gain dynamic ranges.

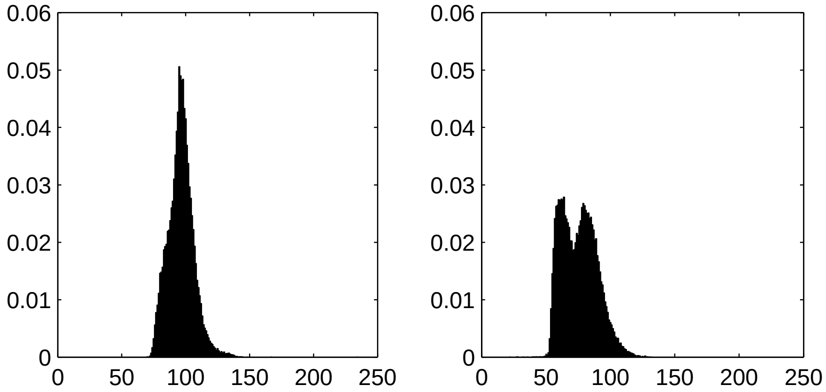
of blocks 1 and 4, and in Fig. 4.21 we have the histograms of blocks 6L and 6H. The latter look alike.

We also take into consideration that both bands 6L and 6H are highly correlated - see the correlations between bands in Table 4.6. This special algorithm is not based on the wavelet transform; instead, the coefficients of the image are simply predicted and the difference is encoded.

In their lossless image compressor LOCO-I [15], Weinberger, Seroussi and Sapiro used what they called the 'median edge detector' (MED) as a fixed predictor for pixel x, in the case of a single image- see Figure 4.22. This median edge detector was formerly used in the work of Martucci [40]. The MED predicts the value of x with $\hat{x}$ according to the formula:

$$\hat{x} = median\{a, b, a + b - c\}.$$

Actually, the MED predicts x with a when it detects a horizontal edge above pixel x, it predicts x with b when it detects a vertical edge left of pixel x, and it predicts x by performing a linear interpolation at the positions of a,



a) b)

Figure 4.20: Histograms of a) block 1 and b) block 4, for a sample stack of blocks.

	band 1	band 2	band 3	band 4	band 5	band 6.1	band 6.2	band 7
band 1	1							
band 2	0,9782	1						
band 3	0,9743	0,9841	1					
band 4	0,7628	0,8052	0,7555	1				
band 5	0,804	0,8014	0,8249	0,6225	1			
band 6.1	0,255	0,2485	0,3103	-0,0715	0,387	1		
band 6.2	0,2612	0,2539	0,3155	-0,0616	0,3928	0,9737	1	
band 7	0,8826	0,8761	0,9083	0,5817	0,9252	0,4659	0,4703	1

Table 4.6: Interband correlations for image San Luis.

b and c in all the other cases.

For the compression of a Landsat image, specifically for the compression of band 6L, which is compressed after band 6H, we aimed at a predictor similar to MED, but which would also include the information of pixel z (same position as x in the previous band 6H)- see Figure 4.23. This revealed itself insufficient. Histograms from band 6L and 6H, however similar in shape, have different mean values and different standard deviations. It was necessary to equalize band 6H to band 6L, by matching their cumulative histograms, before we made use of pixel z.

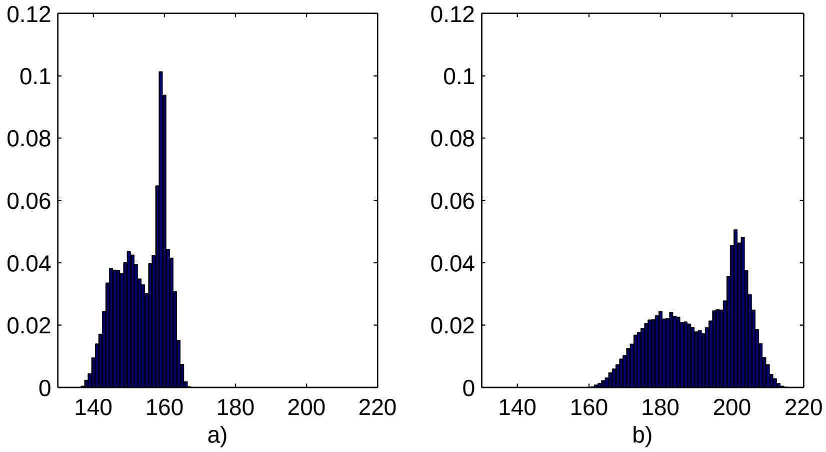


Figure 4.21: Histograms of a) block 6L and b) block 6H for a sample stack of blocks.

c	b
a	x

Figure 4.22: One image: pixels for prediction of x.

Consider band 6H as a sample of random variable U, with cumulative distribution F_U, and consider band 6L as a sample of random variable Y, with cumulative distribution F_Y. For each value u, let v be the value such that

$$F_Y(v) = F_U(u)$$

and define $g(u)$ as

$$g(u) = v = F_Y^{-1} F_U(u);$$

then $g(U)$ has the desired cumulative distribution similar to F_Y [41].

	z

6H

c	b
a	x

6L

Figure 4.23: Landsat image: pixels for the prediction of x in band 6L.

c	b	d
a	x	

Figure 4.24: Pixels for block 6L gradient template.

By calculating $g(u)$ for each pixel u of block 6H (in the stack being processed), we transform block 6H to a new image having a cumulative histogram similar to that of block 6L: the values will be akin to the values of block 6L, and useful for predicting them. Accordingly, the proposed prediction of our method for pixel x in block 6L is

$$\hat{x} = median\{a, b, a + b - c, g(z)\}.$$

The predictions errors $x - \hat{x}$ were encoded with an entropy-based coder. The size of the whole band 6L of the San Luis image was thus reduced from 50.2 MB to 5.5 MB. Further reduction was obtained by using contexts in the entropy-based coder. As in LOCO-I [15], the contexts used were the gradients of each pixel of band 6L, that is, $d - b$, $b - c$ and $c - a$ (see Figure 4.24). To decrease the great number of possible contexts, the values of the gradients were grouped into three classes $\{0\}$, $\{-1, 1\}$ and $\{x \in \mathbb{Z} : |x| > 1\}$. Band 6L finally occupied 5.1 MB, after the context-based entropy coder was run. This gives a compression ratio of 9.84:1 with no loss of information.

Chapter 5

Hyperspectral image compression

5.1 State of the art for lossless compression of hyperspectral images

Most of the hyperspectral compressors are prediction–based. Since spectral correlation is usually high (much more higher than spatial) pixels are predicted with others in an adjacent band (rather than pixels surrounding the one to be predicted). SLSQ [36], a low-complexity method designed for hyperspectral image compression, performs a simple prediction for each pixel, by taking a constant times the same pixel in the previous band. The constant is calculated by least squares over 4 previously encoded neighboring pixels. SLSQ–OPT version of SLSQ performs one band look–ahead to determine if the whole band is better compressed this way or with intraband prediction, while in the SLSQ–HEU version this decision is taken by an offline heuristic. CCAP [42] predicts a pixel with the conditional expected value of a pixel given the context. The expected value is calculated over coded pixels having matching (highly correlated) contexts. Slyz and Zhang

[43] propose 2 compressors (BH and LM) for hyperspectral images. BH predicts a block as a scalar times the same block in the previous band. Coding contexts are defined by the quantized average error. LM predicts a pixel by choosing among different intraband predictions the one that works best for several pixels at the same position in previous bands.

Mielikainen and Toivanen proposed C-DPCM [44], a method that classifies the pixels at the same location and through all the bands, with vector quantization. Interband prediction is performed using the pixels at the same position in 20 previous bands. Weights, calculated for each class/ band, are sent into the code, as well as the 2D template with the classes. Aiazzi et al. [45] classify the prediction context of every pixel using fuzzy clustering, and then calculate the weights for each class. For compression of hyperspectral images [46] they divide each band into small blocks (16×16), they calculate weights for interband prediction over each pixel in a block, and then make a fuzzy clustering of the weights obtained for all the blocks. For each pixel a membership degree is computed according to the efficiency of the weights (from different clusters) on a causal context of the pixel. The final prediction for a pixel is obtained by a combination of linear predictions involving weights from different clusters, pondered by the degrees of membership of the pixel to each cluster.

It is worth mentioning that wavelet-based compressors such as JPEG2000 [12] have been successfuly used for lossy compression of multiband images, either hyperspectral [47, 48] or general earth data [49].

5.2 Look-up table-based methods

5.2.1 LUT

The Look Up Table algorithm [50] is a fast compression technique based on predicting a pixel with another pixel from the same band. Which pixel is used for prediction is determined by inspecting the previous band. Suppose we want to predict pixel $I_{x,y}^{(z)}$ where z is the band number. Then we seek on band $z-1$ the pixel with the same intensity as $I_{x,y}^{(z-1)}$ which is nearest to it in a causal neighbourhood. Let $I_{x',y'}^{(z-1)}$ be that pixel. Then, the prediction for pixel $I_{x,y}^{(z)}$ will be $I_{x',y'}^{(z)}$. If no match is found, pixel $I_{x,y}^{(z-1)}$ is the one selected for prediction. In order to speed things up, a look up table data structure is used for searching the pixel on the previous band. With this data structure the algorithm is efficiently implemented as shown in Fig 5.1 for consecutive bands z and $z-1$. Then, the difference $I^{(z)} - P^{(z)}$ between the band and its prediction is entropy coded and this process is repeated for $z = 2, \ldots, 224$.

```
Data: Bands $I^{(z)}$, $I^{(z-1)}$ and table initialized as LUTable($i$) = $i$
Result: Prediction for band $z$: $P^{(z)}$
for every pixel $I_{x,y}^{(z)}$ do
    $P_{x,y}^{(z)} \leftarrow$ LUTable($I_{x,y}^{(z-1)}$);
    LUTable($I_{x,y}^{(z-1)}$) $\leftarrow I_{x,y}^{(z)}$;
end
```

Figure 5.1: Look-up Table algorithm.

In Fig. 5.2 entropy values are ploted for each band of the Jasper Ridge image. In dashed line, entropies of the pixels of the image are ploted. 6 steps of the 2D S+P wavelet transform were computed and the entropy of the coefficients is plotted in dotted line for each band. Finally it is shown in gray line the entropy of the prediction differences for the LUT algorithm. It is remarkable how high is the compression achieved with this simple algorithm, which is only based on indexing and updating a table. It is entirely based on the premise of high correlation between bands and designed in order to take advantage of this fact.

Original	Wavelet S+P	LUT prediction
8.6656	6.6587	4.9504

Table 5.1: Average entropies for first scene of Jasper Ridge.

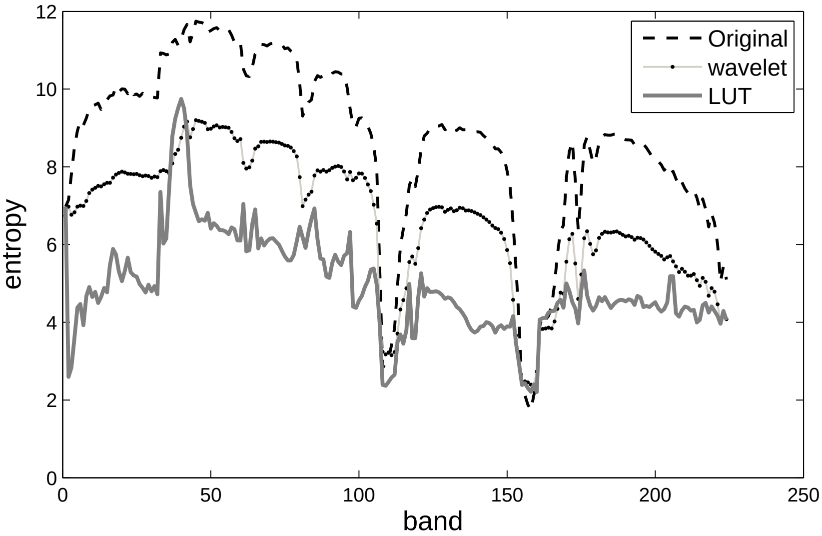


Figure 5.2: Entropy values for every band of the Jasper Ridge image computed over the original image (dashed), over the prediction differences for the LUT method (gray), and over the 2D S+P wavelet transform (dotted). See averaged values in Table 5.1.

5.2.2 LAIS-LUT

An improvement over the LUT algorithm has been presented by Bormin Huang [51]. It was named LAIS-LUT after Locally Averaged Interband Scaling LUT and it is supposed to behave more accurately in presence of outliers. This modification adds an extra LUT table and the predictor is selected from one of the two LUTs. Using a scaling factor α which is precomputed on a causal neighbourhood, an estimate $\tilde{P}^{(z)}_{x,y}$ is calculated for a current pixel $I^{(z)}_{x,y}$ as $\tilde{P}^{(z)}_{x,y} = \alpha I^{(z-1)}_{x,y}$. Since two values are now possible candidates for prediction (one for each LUT), the one that is closer to $\tilde{P}^{(z)}_{x,y}$ is selected as the final prediction (Fig. 5.3 shows LAIS-LUT algorithm). When prediction $P^{(z)}$ for band z is estimated, the prediction error $I^{(z)} - P^{(z)}$ is entropy coded. This is repeated for $z = 2, \ldots, 224$. Notice that when the tables are not initialized, $P_{x,y}^{(z)}$ is selected for prediction, and, the value of the first LUT is used for prediction only when this table is initilialized.

```
Data: Bands $I^{(z)}$, $I^{(z-1)}$, tables LUTable$_1$ and LUTable$_2$ are not initialized
Result: Prediction for band $z$: $P^{(z)}$
for every pixel $I_{x,y}^{(z)}$ do
    if both tables are not initialized on entry (x,y) then
        $P_{x,y}^{(z)} \leftarrow \alpha I_{x,y}^{(z-1)}$
    else if only LUTable$_1$ is initialized in (x,y) then
        $P_{x,y}^{(z)} \leftarrow \text{LUTable}_1(I_{x,y}^{(z-1)})$;
    else
        $P_{x,y}^{(z)} \leftarrow$ closer value of $\{\text{LUTable}_1(I_{x,y}^{(z-1)}), \text{LUTable}_2(I_{x,y}^{(z-1)})\}$ to $\alpha I_{x,y}^{(z-1)}$;
    end
    $\text{LUTable}_2(I_{x,y}^{(z-1)}) \leftarrow \text{LUTable}_1(I_{x,y}^{(z-1)})$;
    $\text{LUTable}_1(I_{x,y}^{(z-1)}) \leftarrow I_{x,y}^{(z)}$;
end
```

Figure 5.3: LAIS-LUT algorithm.

5.2.3 LAIS-QLUT

Mielikainen and Toivanen proposed a modification to the LAIS-LUT method in order to shrink Lookup tables [52]. For that, the value used for indexing into the LUTs is quantized and smaller LUTs can be used (i.e., the value x used as an index in the LUT is replaced by $\lfloor x/q \rfloor$ for some quantization step q, being $\lfloor \cdot \rfloor : \mathbb{R} \to \mathbb{Z}$ a function that maps to a close integer). In the previous section, LAIS-LUT found exact matches in the previous band in order to obtain the predictor in the current band. Now, this search is no more exact and the predictor's selection in the current band is based on 'similarities' in the previuos band. In the tests performed here, the same quantization step is used for each band. Resuls for the Jasper Ridge image are presented in Table 5.2 and in Figure 5.4 for several quantization steps.

q	10	20	25	28	30	35
entropy	4.6772	4.7761	4.8411	4.8805	4.9075	4.9730

Table 5.2: Mean entropies using the corresponding quantization step q in the LAIS-QLUT method.

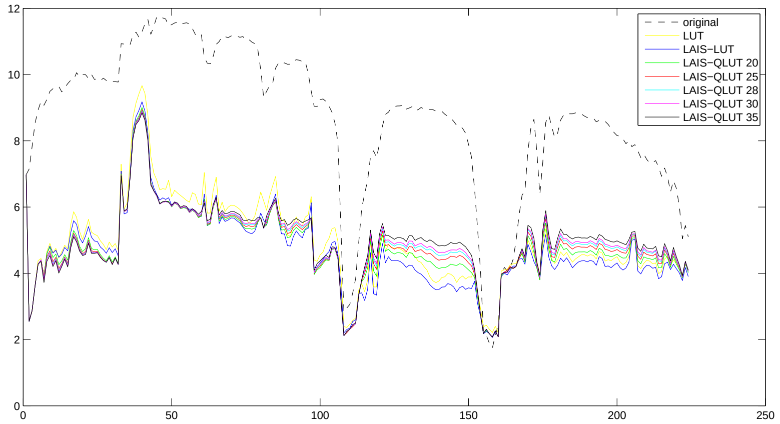


Figure 5.4: Entropies for LUT, LAIS-LUT, and LAIS-QLUT methods, for $q = 20, 25, 28, 30, 35$.

5.2.4 LAIS-LUT + Classification

In this subsection a classification step is added before the LAIS-LUT algorithm is applied. For that, LAIS-LUT is applied on pixels belonging to each class separately; this is equivalent to using separate LUT tables for each class (see Fig. 5.5 for a description of this algorithm).

```
Data: Image $I$, $N$=number of classes.
begin
    Classify $I$ into $N$ classes;
    Encode $I^{(1)}$;
    for $z = 2 \dots 224$ do
        for $i = 1 \dots N$ do
            $I_i^{(z)} \leftarrow$ pixels in $I^{(z)}$ that belong to class $C_i$;
            $P_i^{(z)} \leftarrow$ LAIS_LUT($I_i^{(z)}$, $I^{(z-1)}$);
        end
        Encode $I^{(z)} - P^{(z)}$
    end
end
```

Figure 5.5: LAIS-LUT algorithm with classes.

Note that every time the function LAIS_LUT is called, LUT tables are reset. In addition, a template indicating pixel classes should be sent to the decoder, since classification is performed on all bands.

This method did not show any improvements on bitrates.

5.3 LAIS-LUT with Varying Degrees of Confidence (LAIS-LUT-VDC)

Based on observed examples where LAIS-LUT predictions were inaccurate, an improvement was introduced in this thesis, which was partly published in [53]. In LAIS-LUT, when both LUTs entries reference a pixel on the current band (i.e., both tables have been updated at least once), one of the two values is selected, whether it is near to the LAIS prediction $P_{x,y}^{(z)}$ or not. The scaling factor (LAIS) is estimated on quotients between pixels on consecutive bands; these pixels surround spatialy the one to be predicted (see equation 5.2). In order to avoid outliers when predicting, the LAIS-LUT method assumes that either

$$\alpha I_{x,y}^{(z-1)} \approx \text{LUTable}_1(I_{x,y}^{(z-1)}) \quad \text{or} \quad \alpha I_{x,y}^{(z-1)} \approx \text{LUTable}_2(I_{x,y}^{(z-1)}),$$

that is, $\alpha I_{x,y}^{(z-1)}$ is close to the LUT prediction, or, the scaling factor observed in the neighbourhood is similar to the scaling factor between the pair values (index and content) of one of the LUTs. Roughly speaking, the actual pixel and the one used for prediction are implicitly conjectured to belong to the same 'class'. When this is not the case, predictions are inaccurate. For that reason, a distinction may be made based on the proximity of these values. Let $\tilde{t}$ be the lookup table whose $I_{x,y}^{(z-1)}$-th entry is closest to the LAIS-LUT predicted value, ie,

$$\tilde{t} = \underset{1 \le i \le 2}{\arg\min} \left| \text{LUTable}_i(I_{x,y}^{(z-1)}) - \alpha I_{x,y}^{(z-1)} \right|$$

and let $\tilde{P}_{x,y}^{(z)}$ be the prediction, that is, $\tilde{P}_{x,y}^{(z)} = \text{LUTable}_{\tilde{t}}(I_{x,y}^{(z-1)})$.
For a given threshold u_z, the final predictions $P_{x,y}^{(z)}$ for the method LAIS-LUT-VDC (LAIS-LUT with Varying Degrees of Confidence) is defined as

$$P_{x,y}^{(z)}(u_z) = \begin{cases} \tilde{P}_{x,y}^{(z)} & \text{if } \left| \alpha - \frac{\tilde{P}_{x,y}^{(z)}}{I_{x,y}^{(z-1)}} \right| < u_z \\ \alpha I_{x,y}^{(z-1)} & \text{otherwise} \end{cases} \tag{5.1}$$

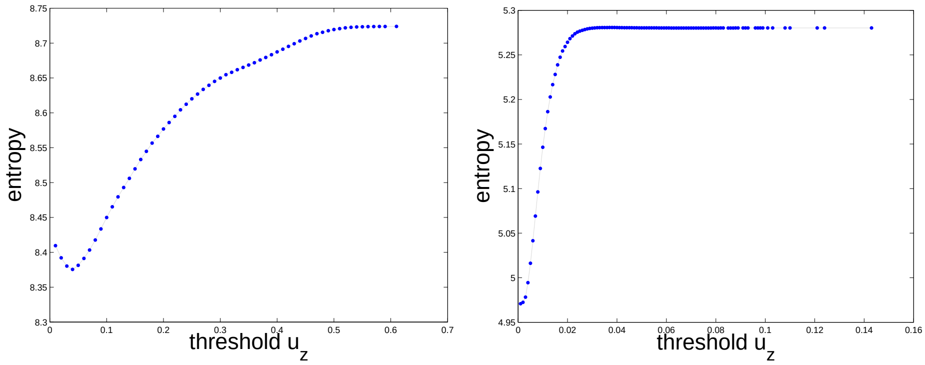
In order to determine the best value for u_z, the entropy of the prediction error is minimized:

$$\min_{u_z} H(I^{(z)} - P^{(z)}(u_z))$$

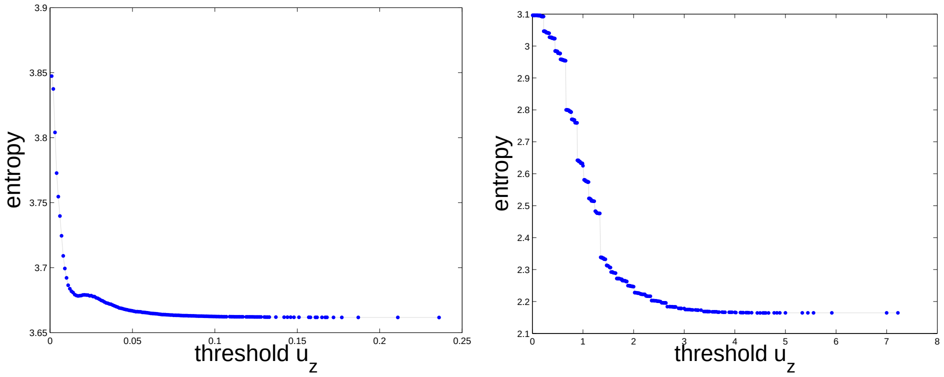
This minimization is done for each band. We used the golden section search [54] algorithm which is quite simple and converges linearly to the minimum. In Fig. 5.6 the entropy of the error prediction is shown for bands 38, 92, 144 and 157. The x-axis of each plot indicates the threshold utilized to estimate the entropy of the prediction error. It should be noticed that this new method is a generalization of LAIS-LUT, since as the threshold becomes larger the prediction chosen according to equation (5.1) is more similar to LAIS-LUT. Fig. 5.6 (c) and (d) shows that LAIS-LUT performs better on bands 144 and 157 since the larger the threshold (and therefore, the method behaves more like LAIS-LUT) the smaller is the entropy; in other cases, (Fig. 5.6 (a) and (b)) finding other thresholds is worth the effort.

Although they may seem smooth functions, they are not if we look closer –see Fig. 5.6 (d) for a clear example. For that reason, faster algorithms (such as Newton) were not used for minimization since they assume continuous functions and, in order to obtain such smooth version, too many evaluations of the entropy had to be performed. The golden-search reaches the minimun in less than 20 iterations, with only one evaluation of the entropy per iteration. Another constraint of this algorithm is that the function to be minimized must be unimodal. Extra care had to be taken on the initialization of this algorithm since it starts on 3 points where the function value in the middlepoint must be less than that of the other 2. We used a recursive algorithm in order to determine these three points; in case these points cannot be found (probably because the function is strictly increasing/decreasing) the minimun value is returned.

Fig. 5.7 shows the LAIS-LUT-VDC algorithm for two consecutive bands. Once the prediction $P^{(z)}$ for band z is estimated, the difference $I^{(z)} - P^{(z)}$



(a) Band 38 (b) Band 92



(c) Band 144 (d) Band 157

Figure 5.6: Entropy of $I^{(z)} - P^{(z)}(u_z)$ depending on threshold u_z, for bands $z = 38, 92, 144, 157$ of Jasper Ridge image. Prediction P is estimated according to LAIS-LUT-VDC algorithm shown in Fig. 5.7.

is entropy coded. This is repeated for $z = 2, \ldots, 224$.

In this modified version of LAIS-LUT, the LUT prediction is trusted only if the quotient between this prediction and the pixel in the previous band is similar to α (i.e., they are in the same 'class'). If not, the surrounding pixels are trusted for predicting. This may be thought of as a division into classes (depending on the threshold): on one side we rely upon the high correlation between the two bands (a global meter), which is the underlying premise of the LUT and LAIS-LUT methods; on the other side, we restrict to the local activity of the pixels examining their behaviour on a confined

```
Data: Bands $I^{(z)}$, $I^{(z-1)}$, threshold $u_z$, tables LUTable$_1$ and
      LUTable$_2$ are not initialized
Result: Prediction for band $z$: $P^{(z)}$
begin
    for every pixel $I^{(z)}_{x,y}$ do
        if both tables are not initialized then
            $P^{(z)}_{x,y} \leftarrow \alpha I^{(z-1)}_{x,y}$;
        else
            // get best LUT prediction
            if only LUTable$_1$ is initialized then
                $\tilde{P}^{(z)}_{x,y} \leftarrow \text{LUTable}_1(I^{(z-1)}_{x,y})$;
            else
                $\tilde{P}^{(z)}_{x,y} \leftarrow$ closer value of
                $\{\text{LUTable}_1(I^{(z-1)}_{x,y}), \text{LUTable}_2(I^{(z-1)}_{x,y})\}$ to $\alpha I^{(z-1)}_{x,y}$;
            end
            // determine final prediction
            if $\left|\alpha - \frac{\tilde{P}^{(z)}_{x,y}}{I^{(z-1)}_{x,y}}\right| < u_z$ then
                $P^{(z)}_{x,y} \leftarrow \tilde{P}^{(z)}_{x,y}$;
            else
                $P^{(z)}_{x,y} \leftarrow \alpha I^{(z-1)}_{x,y}$;
            end
        end
        $\text{LUTable}_2(I^{(z-1)}_{x,y}) \leftarrow \text{LUTable}_1(I^{(z-1)}_{x,y})$;
        $\text{LUTable}_1(I^{(z-1)}_{x,y}) \leftarrow I^{(z)}_{x,y}$;
    end
end
```

Figure 5.7: LAIS-LUT-VDC algorithm for consecutive bands $z-1$ and z.

neighbourhood.

5.3.1 Estimating the scaling factor

We choose among 4 possible ways to estimate α for our method:

1. **Locally Averaged Interband Scale.** This is how α is computed for LAIS-LUT (as described in [51]):
$$\alpha_1 = \frac{1}{3}\left(\frac{I^{(z)}_{x-1,y}}{I^{(z-1)}_{x-1,y}} + \frac{I^{(z)}_{x,y-1}}{I^{(z-1)}_{x,y-1}} + \frac{I^{(z)}_{x-1,y-1}}{I^{(z-1)}_{x-1,y-1}}\right) \tag{5.2}$$
2. **Median of quotients.**
$$\alpha_2 = \text{median}\left\{\frac{I^{(z)}_{x-1,y}}{I^{(z-1)}_{x-1,y}}, \frac{I^{(z)}_{x,y-1}}{I^{(z-1)}_{x,y-1}}, \frac{I^{(z)}_{x-1,y-1}}{I^{(z-1)}_{x-1,y-1}}\right\}$$
3. **Quotient of Local Sums.** Proposed in [52] for the LAIS-QLUT method:
$$\alpha_3 = \frac{I^{(z)}_{x-1,y} + I^{(z)}_{x,y-1} + I^{(z)}_{x-1,y-1}}{I^{(z-1)}_{x-1,y} + I^{(z-1)}_{x,y-1} + I^{(z-1)}_{x-1,y-1}}$$
4. **Least Squares.** This is the least squares estimator for the model $\mathbf{I}^{(z-1)}_{x,y} \cdot \alpha = \mathbf{I}^{(z)}_{x,y}$ for three neigbouring pixels:
$$\alpha_4 = \frac{I^{(z)}_{x-1,y}I^{(z-1)}_{x-1,y} + I^{(z)}_{x,y-1}I^{(z-1)}_{x,y-1} + I^{(z)}_{x-1,y-1}I^{(z-1)}_{x-1,y-1}}{(I^{(z-1)}_{x-1,y})^2 + (I^{(z-1)}_{x,y-1})^2 + (I^{(z-1)}_{x-1,y-1})^2}$$

We made tests in order to find the best estimate for α among the four choices given. According to results given in Table 5.3, on average better results were obtained with α_3. This is the estimate for α that we chose for our method.

5.3.2 Results

Bitrates results for each band are shown in Figs. 5.8 and 5.9 for Jasper Ridge and Low Altitude images respectively. For comparison, we extract from [52] and [55] compression ratio results for other lossless methods and

	α_1	α_2	α_3	α_4
Jasper	3.4866	3.4659	**3.4919**	3.4913
Low Alt	3.2404	3.2218	3.2461	**3.2475**
Cuprite	3.6669	3.6539	**3.6701**	3.6683
Lunar	3.5456	3.5330	**3.5493**	3.5473
Average	3.4849	3.4686	3.4894	3.4886

Table 5.3: Compression ratios obtained by our method LAIS-LUT-VDC using different estimations of the scaling factor.

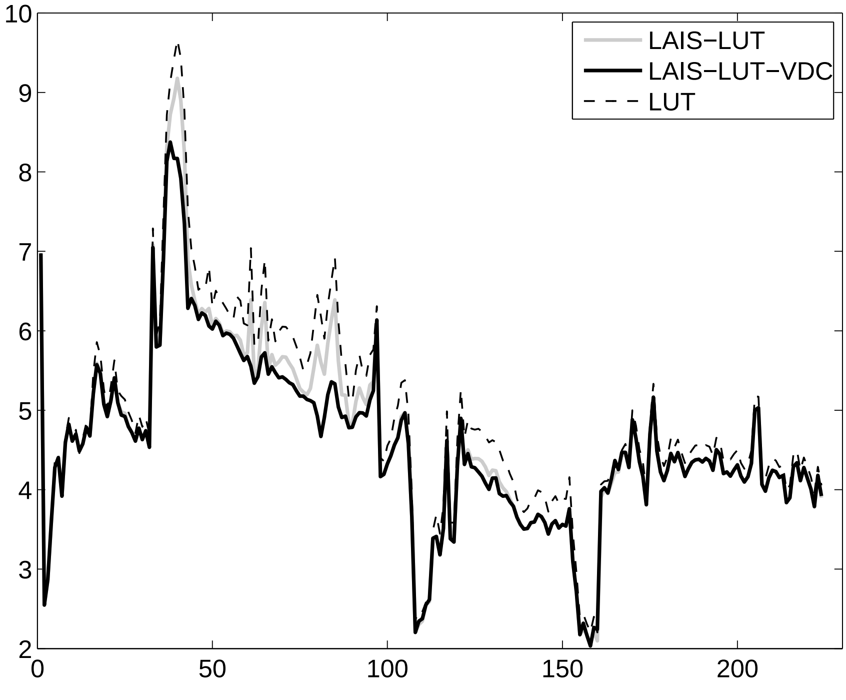


Figure 5.8: Bitrates per band for Jasper Ridge image.

append our achieved compression ratios in the last row (see Tables 5.4 and 5.5). BH, SLSQ, CCAP and C-DPCM were introduced in Section 5.1. 3D-CALIC [20] and M-CALIC [56] are CALIC-based [9] compressors developed for multiband images. NPHI [57] is a method for hyperspectral image compression that predicts pixels from its context (in the current and previous band) and EPHI extends NPHI using edge-based analysis. Finally,

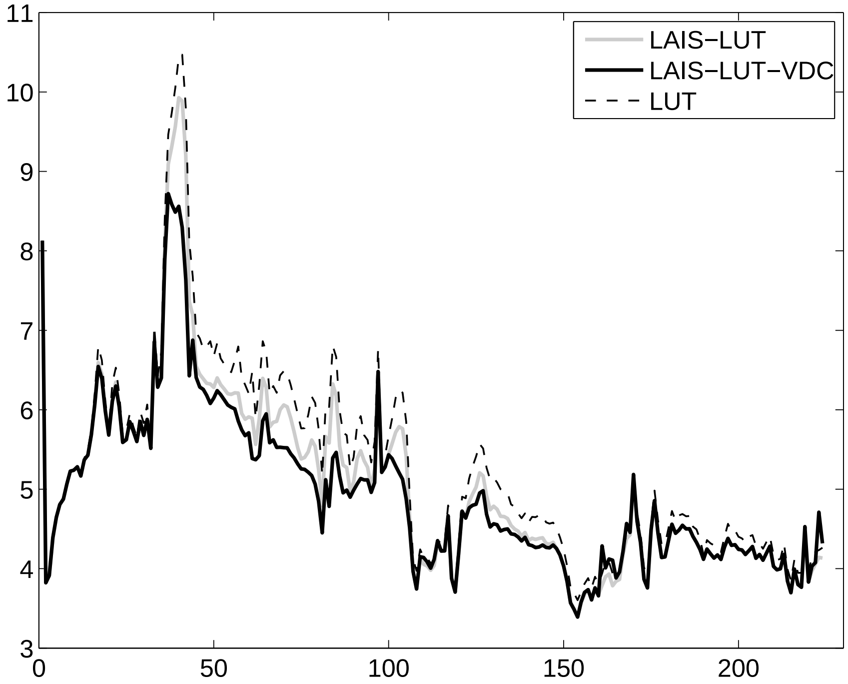


Figure 5.9: Bitrates per band for Low Altitude image.

Algorithm	Cuprite	Jasper Ridge	Lunar Lake	Low Alt
3D-CALIC	2.97	2.98	3.01	
SLSQ	3.15	3.15	3.15	
M-CALIC	3.14	3.06	3.19	
SLSQ-HEU	3.23	3.22	3.23	
NPHI	3.34	3.27	3.22	
LUT	3.44	3.23	3.4	
S-RLP	3.41	3.44	3.41	
S-FMP	3.43	3.46	3.43	
RKLT ML 4+JPEG2K	3.30	3.31	3.23	3.09
LAIS-LUT	3.58	3.42	3.53	
LAIS-LUT-VDC	3.67	3.49	3.55	3.25

Table 5.4: Compression ratios for BIL methods.

Algorithm	Cuprite	Jasper Ridge	Lunar Lake	Low Alt
CCAP	3.53	2.56	3.21	
BH	3.13	3.06	3.13	
3D-CALIC	3.06	3.08	3.09	
SLSQ-OPT	3.24	3.23	3.23	
M-CALIC	3.22	3.17	3.28	
LUT	3.44	3.23	3.4	
EPHI	3.36	3.29	3.28	
C-DPCM	3.42	3.46	3.37	
S-RLP	3.41	3.44	3.41	
S-FMP	3.43	3.46	3.43	
RKLT ML 4+JPEG2K	3.30	3.31	3.23	3.09
LAIS-LUT	3.58	3.42	3.53	
LAIS-LUT-VDC	3.67	3.49	3.55	3.25

Table 5.5: Compression ratios for BSQ methods.

the method RKLT ML 4 + JPEG2K [55] applies a 4-level multiclustered reversible Karhunen-Loêve transform in the spectral dimension; the spectral decorrelated data is then losslessly coded with JPEG2000. Results are reported for BSQ (band sequential) and BIL (band interleaved by line) formats. Notice that look-up table methods do not benefit from any particular format since only data from the line in the previous band is needed and no more than one band is used by the methods. In the results provided, our method outperforms LUT and LAIS-LUT.

5.4 LAIS-LUT-VDC with classes in the context

The scaling factor used in LAIS-LUT-VDC, is computed in a small neighbourhood of the pixel to be predicted using just three pixels so as to be as fast as possible. This neighbourood may be expanded in order to include more pixels. However, pixels that are not related with the one to be predicted may introduce a bias in the estimation of the scaling factor. If, on the contrary, this pixels are related (i.e., are in the same class) the estimation of the scaling factor would still be accurate. That is why, in order to make LAIS-LUT-VDC predictions more accurate, in this approach we try to improve the prediction that does not come from the lookup tables. For that, the scaling factor is enhanced by:

1. expanding the context size in which it is calculated;
2. only using the pixels that are in the same class with respect to the one to be predicted.

Two ways of obtaining a template of classes were used:

5.4.1 Entropy-based classification.

AVIRIS images are formed by 224 bands. Each pixel $I_{x,y}$ may be thought of as a vector in $\mathbb{Z}^{224}$ where each component belongs to a different band. In Figure 5.10 the first scene of the Jasper Ridge image is shown along with four pixels belonging to different classes, marked with symbols '$\oplus$', '$\circ$', '$\heartsuit$' and '$\diamondsuit$'. The spectral vectors associated to the same 4 positions are plotted in Figure 5.11, showing different behaviours depending on what type of soil is being considered (this was mentioned earlier when spectral signatures were introduced).

In order to classify the image, for every pixel, a 1-D wavelet transform is applied along the spectral direction. Then, the entropy of each transformed spectral vector is computed, giving an image $\mathcal{S}$ where $S_{x,y} := H(w(I_{x,y}))$ where $H(\cdot)$ denotes the entropy function and w is a 1-D wavelet transform. In Figure 5.12 a grayscale image of the entropies are displayed. This image may be further split into classes with an unsupervised classifier (in this work,

Figure 5.10: Band 50 of Jasper Rigde image.

K-means in 1 dimension was used). Results of compressing AVIRIS images with LAIS-LUT-VDC, in which the scaling factor is calculated with pixels belonging to the same class, and the classification is carried out with the mentioned entropy-based method are given in Table 5.6 (a) - bitrates are shown for different sizes of the context and number of classes.

5.4.2 Classification based on selected bands

In Figure 5.11 it can be observed that low reflectance values are present in some groups of bands, regardless of the landscape to which the pixels belong. In most bands, though, the spectral behaviour is different according to the landscape, and this can be useful for classification. In this second approach, a few of these bands are selected for unsupervised classification. They are bands 45, 70, 93, 175, 183. In Figure 5.11 they are indicated with black vertical lines. K-means in 5 dimensions was implemented. Results of com-

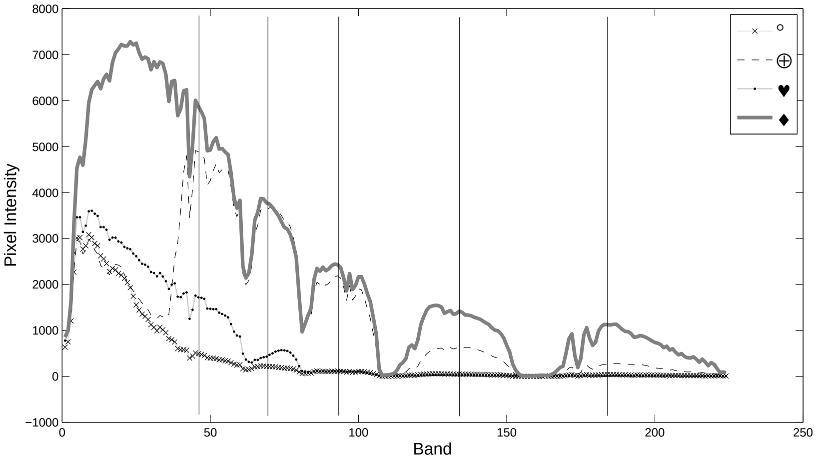


Figure 5.11: Spectral signatures of different classes: values of selected pixels in Fig 5.10 throughout the bands. Vertical lines indicate the bands chosen for K-means classification.

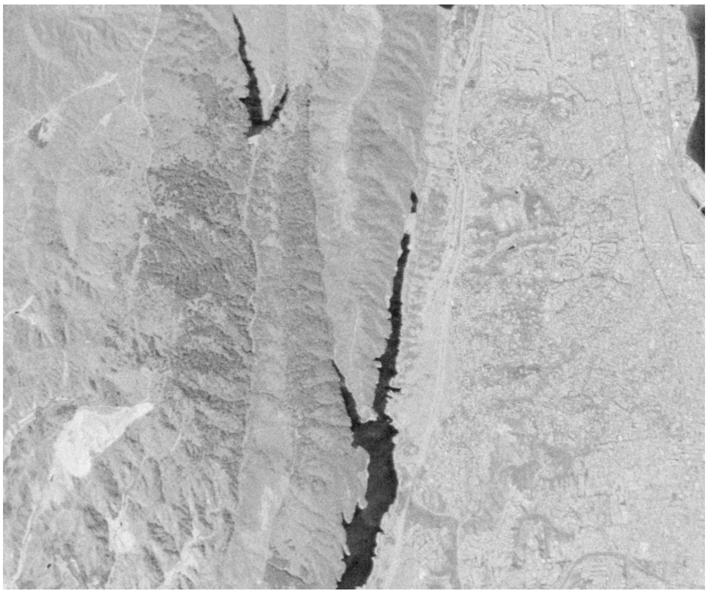


Figure 5.12: Entropy of the wavelet transform of every pixel.

pressing AVIRIS images with LAIS LUT-VDC, in which the scaling factor is calculated with pixels belonging to the same class, and the classification is carried out with K-means on 5 selected bands, are listed in Table 5.6 (b)-bitrates are shown for different sizes of the context and number of classes.

Image	2 classes					3 classes				
	CZ 4	CZ 5	CZ 6	CZ 7	CZ 8	CZ 4	CZ 5	CZ 6	CZ 7	CZ 8
Cuprite	4.3285	4.3181	4.3112	4.3140	4.3094	4.3348	4.3247	4.3182	4.3140	4.3094
Jasper	4.5926	4.5883	4.5877	4.6125	4.6124	4.5998	4.5925	4.5891	4.6125	4.6124
Low Alt	4.9412	4.9368	4.9371	4.9734	4.9765	4.9582	4.9496	4.9463	4.9734	4.9765
Lunar	4.4665	4.4598	4.4562	4.4593	4.4554	4.4793	4.4699	4.4643	4.4593	4.4554
Avg	4.5822	4.5757	4.5731	4.5898	4.5884	4.5930	4.5842	4.5795	4.5898	4.5884

(a) Classes determined with the entropy-based classification.

Image	2 classes					3 classes				
	CZ 4	CZ 5	CZ 6	CZ 7	CZ 8	CZ 4	CZ 5	CZ 6	CZ 7	CZ 8
Cuprite	4.3203	4.3088	4.3020	4.2973	4.2952	4.3493	4.3296	4.3168	4.3104	4.3065
Jasper	4.5936	4.5931	4.5939	4.5957	4.5978	4.5956	4.5891	4.5864	4.5855	4.5862
Low Alt	4.9480	4.9409	4.9392	4.9400	4.9425	4.9490	4.9420	4.9396	4.9401	4.9422
Lunar	4.4686	4.4565	4.4492	4.4456	4.4429	4.4956	4.4773	4.4664	4.4593	4.4554
Avg.	4.5826	4.5748	4.5711	4.5697	**4.5696**	4.5974	4.5845	4.5773	4.5738	4.5726

(b) Classes determined with K-means on selected bands.

Table 5.6: Bitrate results for LAIS-LUT-VDC with the scaling factor estimated on the context using pixels of the same class. CZ stands for Context Size.

Chapter 6

Conclusions

In this thesis the lossless compression of satellite images has been addressed. Two major types of multiband images were considered: multispectral and hyperspectral.

For multispectral images a new lossless compressor that exploits both spatial and spectral correlation was presented. With an integer wavelet transform a significant reduction of spatial correlation was obtained. To reduce spectral correlation (and remaining spatial correlation) we rely on inter and intraband predictions and encoding of prediction differences. These are performed on the wavelet coefficients instead of the pixels. Tests for different models of statistical dependencies between wavelet detail coefficients, showed the importance of separating information into classes. The idea of conditioning the estimation of weights to the class (besides the band, block and subband), has led to more accurate predictions. This resulted in lower entropies of prediction differences. Spectral correlations were exploited in three more ways. In each stack of blocks, correlations between wavelet–transformed bands determined a new band order. In the classification step, initial classes for a Lloyd–Max quantizer were set according to the classes of the block in the previous band. And pixel classes (and induced classes for wavelet coefficients) were used to estimate the classes of the block in the next band.

Hyperspectral images have quite different characteristics compared to multispectral images. In case of AVIRIS, their high spectral resolution make them suitable for spectral prediction; on the other hand, 2D transforms cannot achieve a good spatial decorrelation because of the high radiometric resolution (bitdepth) of these images. An algorithm based on look-up tables was presented. The simple and efficient LAIS-LUT compressor fails to predict accurately in some cases. The prediction in LAIS-LUT is necessarily obtained by one of two lookup tables. These tables keep track of pixel values that, have the same intensity with respect to the one to be predicted (if we restrict to the previous band). So if these pixels in the previous band have the same intensity, the corresponding pixels (ie, at the same location) in the current band are supposed to have the same (or similar) values too (because of the high correlation between adjacent bands). Those pixels in the current band are: the one to be predicted, and the one kept in the table that is used for prediction. So, values in the tables are used for prediction. A scaling factor, computed on a surrounding neighbourhood of the pixel to be predicted, is used only to determine which one of the two tables are used. The scaling factor itself may be used to generate a prediction as well. By setting a threshold value that allows to either predict with a lookup table value or by using the scaling factor, a more flexible version of LAIS-LUT was designed: LAIS-LUT-VDC. This may be thought as a division into classes (depending on the threshold): on one side we rely upon the high correlation between the two bands (a global meter), which is the underlying premise of the LUT and LAIS-LUT methods; on the other side, we restrict to the local activity of the pixels examining their behaviour on a confined neighbourhood. A classification step has been added as well in order to compute the scaling factor in a more suitable way. This classification is performed considering all the bands and determining a class for each pixel. By extending the neighbourhood in which the scaling factor is computed and only allowing the utilization of those pixels in the neighbourhood that have the same class of that to be predicted, a more precise scaling factor is obtained. This in turn, favors more accurate predictions in LAIS-LUT-VDC since the scaling factor is a key feature: it is used, depending on the threshold, either to generate a prediction or to select a value from one of the lookup tables.

For the design of these two lossless compressors, different properties of the images were studied thoroughly, namely, their resolutions (spectral, spatial and radiometric) and correlations (spatial and spectral). Techniques and algorithms which take advantage of each set of properties have been presented. Original contributions of this work, are: an analysis of different models for wavelet coefficient's dependencies, a novel combination of techniques for the compression of multispectral images, an original method for band ordering, based on wavelet coefficients correlation, an original variant of the LAIS-LUT method for the compression of hyperspectral images, and an original segmentation of these images based on the entropy of 1D wavelet coefficients, calculated along the spectral direction. All of these contributions aim at achieving competitive compression rates for the lossless compression of these satellite images.

Bibliography

[1] F. F. Sabins, *Remote Sensing: Principles and Interpretation*, 3rd ed. Waveland Press, 2007.

[2] J. B. Campbell, *Introduction to Remote Sensing*, 4th ed. Guilford Press, 2007.

[3] NASA, *The Landsat Program, http://landsat.gsfc.nasa.gov/.*

[4] U. G. Service, *Landsat Missions, http://landsat.usgs.gov/.*

[5] NASA, *Landsat 7 Science Data Users Handbook, http://landsathandbook.gsfc.nasa.gov/handbook.html.*

[6] JPL, "Aviris homepage," *http://aviris.jpl.nasa.gov/.*

[7] K. Sayood, *Introduction to Data Compression*, 2nd ed. Morgan Kaufmann Publishers, 2000.

[8] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, pp. 379–423,623–656, 1948.

[9] X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Trans. Communications*, vol. 45, no. 4, pp. 437–444, 1997.

[10] A. R. Calderbank, I. Daubechies, W. Sweldens, and B. Yeo, "Wavelet transforms that map integer to integers," *Applied and Computational Harmonics Analysis*, vol. 5, no. 3, pp. 332–369, 1998.

[11] A. Said and W. Pearlman, “A new fast and efficient image codec based on set partitioning in hierarchical trees,” *IEEE Trans. Circuits Sys. Video Tech.*, vol. 6, pp. 243–250, 1996.

[12] D. Taubman and M. Marcellin, *JPEG2000: Image compression fundamentals, standards and practice.* Kluwer Academic Publishers, Boston, 2002.

[13] A. Skodras, C. Christopoulos, and T. Ebrahimi, “The JPEG 2000 still image compression standard,” *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, September 2001.

[14] M. Rabbani and R. Joshi, “An overview of the JPEG 2000 still image compression standard,” *Sig. Proc. Im. Comm.*, vol. 17, pp. 3–48, 2002.

[15] M. Weinberger, G. Seroussi, and G. Sapiro, “The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS,” *IEEE Trans. on Image Processing*, vol. 9, pp. 1309–1324, 2000.

[16] G. Motta, J. A. Storer, and B. Carpentieri, “Lossless image coding via adaptive linear prediction and classification,” *Proceedings of the IEEE*, vol. 88, no. 11, pp. 1790–1796, 2000.

[17] M. Slyz and D. Neuhoff, “A nonlinear VQ–based predictive lossless image coder,” *IEEE Proc. Data Compression Conf.*, pp. 304–310, 1994.

[18] X. Tang, S. Cho, and W. A. Pearlman, “Comparison of 3D set partitioning methods in hyperspectral image compression featuring an improved 3D-SPIHT,” *Proc. Data Compression Conf.*, pp. 449–449, 2003.

[19] H. S. Lee, N.-H. Younan, and R. L. King, “Hyperspectral image cube compression combining JPEG 2000 and spectral decorrelation,” *Proc. IEEE IGARSS 2000*, vol. 6, pp. 3317–3319, 2000.

[20] X. Wu and N. Memon, “Context-based lossless interband compression –extending CALIC,” *IEEE Tr. Imag. Proc.*, vol. 9, pp. 994–1001, 2000.

[21] K. Yang and A. Faryar, “A contex-based predictive coder for lossless and near-lossless compression of video,” *Proceedings of International Conference on Image Processing*, vol. 1, pp. 144–147, 2000.

[22] J. Wang, M. Zhang, and S. Tang, “Spectral and spatial decorrelation of Landsat-TM data for lossless compression,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 33, no. 5, pp. 1277–1285, 1995.

[23] R. Buccigrossi and E. Simoncelli, “Image compression via joint statistical characterization in the wavelet domain,” *IEEE Trans Signal Proc*, vol. 8, pp. 1688–1701, 1999.

[24] J. Liu and P. Moulin, “Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients,” *IEEE Tra. Im. Proc.*, vol. 10, pp. 1647–1658, 2001.

[25] A. Benazza-Benyahia, J. Pesquet, and M. Hamdi, “Vector-lifting schemes for lossless coding and progressive archival of multispectral images,” *IEEE Tr. Geo. Rem. Sens.*, vol. 40, no. 9, pp. 2011–2024, 2002.

[26] G. Wallace, “The jpeg still picture compression standard,” *IEEE Trans. Consumer Electronics*, vol. 38, 1992.

[27] A. Said and W. Pearlman, “An image multiresolution representation for lossless and lossy compression,” *IEEE Tr. Im. Proc.*, vol. 5, no. 9, 1996.

[28] A. Ruedin and D. Acevedo, “Prediction of coefficients for lossless compression of multispectral images,” *Proceedings of SPIE*, vol. 5889, pp. 202–211, 2005.

[29] D. Acevedo and A. Ruedin, “Reduction of interband correlation for Landsat image compression,” *XXIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, pp. 390–396, 2005.

[30] E. Chuvieco, *Fundamentos de teledetección espacial.* RIALP, 1996.

[31] A. Ruedin and D. Acevedo, “A class-conditioned lossless wavelet-based predictive multispectral image compressor,” *IEEE Geoscience and Remote Sensing Letters*, vol. 7, pp. 166 – 170, January 2010.

[32] S. Lloyd, “Least squares quantization in PCM,” *IEEE Trans. on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[33] S. R. Tate, “Band ordering in lossless compression of multispectral images,” *IEEE Trans. Computers*, vol. 46, no. 4, pp. 477–483, 1997.

[34] D. Acevedo and A. Ruedin, “A lossless wavelet-based predictive multispectral image compressor,” *Data Compression Conference (DCC2008)*, p. 501, 2008.

[35] W3C, “PNG (portable network graphics) specification,” 2003. [Online]. Available: http://www.w3.org/TR/PNG.

[36] F. Rizzo, B. Carpentieri, G. Motta, and J. A. Storer, “Low complexity lossless compression of hyperspectral imagery via linear prediction,” *IEEE Signal Processing Letters*, vol. 12, no. 2, pp. 138–141, 2005.

[37] A. Krogh, R. Palmer, and J. Hertz, *Introduction to the Theory of Neural Computation.* Santa Fe Institute Editorial Board, 1990.

[38] S. Haykin, *Neural Networks A Comprehensive Foundation.* Prentice Hall, 1999.

[39] M. F. Moller, “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural Networks*, vol. 6, pp. 525–533, 1993.

[40] S. Martucci, "Reversible compression of HDTV images using median adaptive prediction and arithmetic coding," *IEEE Proc. Internat. Symp. on Circuits and Systems*, pp. 1310–1313, 1990.

[41] Gonzalez and Woods, *Digital Image Processing.* Prentice Hall, 2002.

[42] H. Wang, D. Babacan, and K. Sayood, "Lossless hyperspectral image compression using context-based conditional averages," *IEEE Proc. Data Compression Conference*, 2005.

[43] M. Slyz and L. Zhang, "A block-based inter-band lossless hyperspectral image compressor," *IEEE Proc. Data Compression Conf.*, 2005.

[44] J. Mielikainen and P. Toivanen, "Clustered DPCM for the lossless compression of hyperspectral images," *IEEE Trans. Geos. Remote Sensing*, vol. 41, no. 12, pp. 2943–2946, 2003.

[45] B. Aiazzi, P. Alba, L. Alparone, and S. Baronti, "Lossless compression of multi/hyperspectral imagery based on a 3-D fuzzy prediction," *IEEE Trans. Geos. Remote Sensing*, vol. 37, no. 5, pp. 2287–2294, 1999.

[46] B. Aiazzi, L. Alparone, S. Baronti, and C. Lastri, "Crisp and fuzzy adaptive spectral predictions for lossless and near-lossless compression of hyperspectral imagery," *IEEE Geos. Remote Sensing Letters*, vol. 4, no. 4, pp. 532–536, 2007.

[47] J. E. Fowler and J. T. Rucker, "3d wavelet-based compression of hyperspectral imagery," in *Hyperspectral Data Exploitation: Theory and Applications*, C.-I. Chang, Ed. John Wiley & Sons, Inc., 2007, pp. 379–407.

[48] I. Blanes and J. Serra-Sagrista, "Cost and scalability improvements to the karhunen-loêve transform for remote-sensing image coding," *Geo-*

science and Remote Sensing, IEEE Transactions on, vol. 48, no. 7, pp. 2854 –2863, july 2010.

[49] P. Kulkarni, A. Bilgin, M. Marcellin, J. Dagher, J. Kasner, T. Flohr, and J. Rountree, "Compression of earth science data with jpeg2000," in *Hyperspectral Data Compression*, G. Motta, F. Rizzo, and J. A. Storer, Eds. Springer US, 2006, pp. 347–378.

[50] J. Mielikainen, "Lossless compression of hyperspectral images using lookup tables," *IEEE SPL*, vol. 13, no. 3, pp. 157–160, 2006.

[51] B. Huang and Y. Sriraja, "Lossless compression of hyperspectral imagery via lookup tables with predictor selection," *Proceedings of SPIE*, vol. 6365, 2006.

[52] J. Mielikainen and P. Toivanen, "Lossless compression of hyperspectral images using a quantized index to look-up tables," *IEEE Geos. Remote Sensing Letters*, vol. 5, no. 3, 2008.

[53] D. Acevedo and A. Ruedin, "Lossless compression of hyperspectral images: Look-up tables with varying degrees of confidence," *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 2010)*, pp. 1314 – 1317, 14-19 March 2010.

[54] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C, The Art of Scientific Computing.* Cambridge University Press, 1999.

[55] I. Blanes and J. Serra-Sagrista, "Clustered reversible-klt for progressive lossy-to-lossless 3d image coding," in *Data Compression Conference, 2009. DCC '09.*, march 2009, pp. 233–242.

[56] E. Magli, G. Olmo, and E. Quacchio, "Optimized onboard lossless and near-lossless compression of hyperspectral data using CALIC," *IEEE*

Geoscience and Remote Sensing Letters, vol. 1, no. 1, pp. 21 –25, January 2004.

[57] S. Jain and D. Adjeroh, "Edge-based prediction for lossless compression of hyperspectral images," *Data Compression Conf.*, pp. 153–162, 2007.